



# THE UNIVERSITY *of* EDINBURGH

This thesis has been submitted in fulfilment of the requirements for a postgraduate degree (e.g. PhD, MPhil, DClinPsychol) at the University of Edinburgh. Please note the following terms and conditions of use:

This work is protected by copyright and other intellectual property rights, which are retained by the thesis author, unless otherwise stated.

A copy can be downloaded for personal non-commercial research or study, without prior permission or charge.

This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the author.

The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author.

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given.

# Multi-view Representation Learning for Natural Language Processing Applications

*Nikos Papasarantopoulos*



Doctor of Philosophy  
Institute for Language, Cognition and Computation  
School of Informatics  
University of Edinburgh

2020



# Abstract

The pervasion of machine learning in a vast number of applications has given rise to an increasing demand for the effective processing of complex, diverse and variable datasets. One representative case of data diversity can be found in multi-view datasets, which contain input originating from more than one source or having multiple aspects or facets. Examples include, but are not restricted to, multimodal datasets, where data may consist of audio, image and/or text.

The nature of multi-view datasets calls for special treatment in terms of representation. A subsequent fundamental problem is that of combining information from potentially incoherent sources; a problem commonly referred to as *view fusion*. Quite often, the heuristic solution of early fusion is applied to this problem: aggregating representations from different views using a simple function (concatenation, summation or mean pooling). However, early fusion can cause overfitting in the case of small training samples and also, it may result in specific statistical properties of each view being lost in the learning process.

A plethora of multi-view representation learning methods has been proposed in the literature, with a large portion of them being based on the idea of maximising the correlation between available views. Commonly, such techniques are evaluated on synthetic datasets or strictly defined benchmark setups; a role that, within Natural Language Processing, is often assumed by the multimodal sentiment analysis problem. This thesis argues that more complex downstream applications could benefit from such representations and describes a multi-view contemplation of a range of tasks, from static, two-view, unimodal to dynamic, three-view, trimodal applications.

More specifically, we experiment with document summarisation, framing it as a multi-view problem where documents and summaries are considered two separate, textual views. Moreover, we present a multi-view inference algorithm for the bimodal problem of image captioning. Delving more into multimodal setups, we develop a set of multi-view models for applications pertaining to videos, including tagging and text generation tasks. Finally, we introduce narration generation, a new text generation task from movie videos, that requires inference on the storyline level and temporal context-based reasoning.

The main argument of the thesis is that, due to their performance, multi-view representation learning tools warrant serious consideration by the researchers and practitioners of the Natural Language Processing community. Exploring the limits of multi-view representations, we investigate their fitness for Natural Language Processing tasks and show that they

are able to hold information required for complex problems, while being a good alternative to the early fusion paradigm.

# Lay Summary

The core function of machine learning algorithms is to build (“learn”) models from data, called training data; models that are able to make predictions or decisions on other data, without being explicitly programmed to do so. Given the importance of training data in that process, a large part of the development of machine learning systems is devoted to the input and the way it can be represented and communicated to the algorithms. Effectively encoding and combining characteristics of the input is a complex and challenging problem. The last decades have seen a shift from the manual enumeration of input characteristics to casting representation as a learning problem itself, where the goal is to create a model that can describe inputs with minimal human intervention.

The popularity of machine learning in a vast number of applications has given rise to an increasing demand for the effective representation of complex, diverse and variable inputs. An example of data variability comes from multimodal data, which consist of more than one modalities (e.g. image, audio, text) and pose a challenge for representation learning. Although there are popular and established techniques to generate representations for each of the modalities separately, the problem of combining representations from different modalities is less explored. The set of techniques devised to combine representations from multiple sources (views) of the data is the objective of multi-view learning.

A plethora of multi-view representation learning methods has been proposed in the literature. Commonly, such techniques are evaluated on synthetic datasets or strictly defined benchmark setups; a role that, within Natural Language Processing, is often assumed by the multimodal sentiment analysis problem. This thesis argues that more complex downstream applications could benefit from such representations and describes a multi-view contemplation of a range of tasks, from static, two-view, unimodal to dynamic, three-view, trimodal applications.

More specifically, we experiment with document summarisation, framing it as a multi-view problem where documents and summaries are considered two separate, textual views. Moreover, we present a multi-view algorithm for the problem of image captioning. Delving more into multimodal setups, we develop a set of multi-view models for applications pertaining to videos. Finally, we introduce narration generation, a new text generation task from movie videos, that requires inference on the storyline level and temporal context-based reasoning.

The main argument of the thesis is that, due to their performance, multi-view representation learning tools warrant serious consideration by the researchers and practitioners of the Nat-

ural Language Processing community. Exploring the limits of multi-view representations, we investigate their fitness for Natural Language Processing tasks and show that they are able to hold information required for complex problems, such as image captioning, while being a good alternative to the early fusion paradigm.

# Acknowledgements

A PhD is, in many ways, an apprenticeship. As such, students are directly or indirectly taught not only the intricacies of their field and the inner workings of research, but also perseverance, patience, dealing with success and failure and many more. I was fortunate to have Shay Cohen as my supervisor; I am deeply grateful not only for his continuous support, his insights on my work and his eagerness to help me in any way he can, but also his attitude towards research and science. Moreover, I would like to thank Chunchuan, Esma, Jiangming, Joana, Marco, and all the other members of our research group. I grew as a person and a scientist through our formal and informal meetings.

During my stay in the ILCC, I had the opportunity to work with people whose research has been really inspiring for me. I would like to thank all people involved in the SUMMA project and Helen Jiang for our collaboration and Shashi Narayan, not only for our collaboration, but also his advice. Thank you, Steve Renals and Mirella Lapata for providing valuable feedback during all the stages of this thesis. Finally, I would like to thank my thesis examiners, Lucia Specia and Ivan Titov for their constructive feedback.

I am grateful for having close friends who helped me maintain a stable lifestyle and enjoy life outside the university during the past years. Andreas Makas, Eirini Atmatzidou, Haris Alysandratos, Katherine Alysandratou and Waseem Albahri, thank you all so much! I am also deeply grateful for the time I spent playing music in Edinburgh. Thank you Baharat Collective! I owe special thanks to Haris Stylianakis, not only for his vital help on starting a life in Scotland, but also for the numerous conversations about research, music, and life, before, during, and I hope after the years I spent in Edinburgh.

I would like to especially thank my brother George and my mother Loula, for giving me all the remote support and encouragement I could have hoped for.

Sophia, you know that without your support and patience I could not have even started to think about writing this thesis. I cannot thank you enough. In many ways, this has been an adventure for both of us. I can't wait for the next one 😊.



# Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

*(Nikos Papasarantopoulos)*

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>I</b>
1.1	Contributions . . . . .	8
1.2	Overview . . . . .	9
1.3	Published Work . . . . .	10
<b>2</b>	<b>Background</b>	<b>II</b>
2.1	Representation Learning . . . . .	12
2.2	Multi-view Learning . . . . .	13
2.2.1	Non-correlational approaches to multi-view learning . . . . .	17
2.2.2	Correlational approaches to multi-view learning . . . . .	18
2.3	Multi-view sequence learning . . . . .	29
2.3.1	Sequence Models . . . . .	30
2.3.2	Multi-view Sequence Models . . . . .	33
2.3.3	Non-correlational multi-view sequential approaches . . . . .	34
2.3.4	Correlational multi-view sequential approaches . . . . .	38
<b>3</b>	<b>Multi-view Document Summarisation</b>	<b>43</b>
3.1	Document Summarisation . . . . .	44
3.1.1	Extractive Summarisation . . . . .	45
3.1.2	Abstractive Summarisation . . . . .	46
3.1.3	Evaluation . . . . .	49
3.2	Multi-view Representation Learning for Document Summarisation . . . . .	53
3.2.1	Learning Module . . . . .	54
3.3	Application on Extractive Document Summarisation . . . . .	54
3.3.1	Summarisation Module . . . . .	56
3.3.2	Experimental Setup . . . . .	58
3.3.3	Experiment: Learning Representations with DCCA . . . . .	60

3.3.4	Experiment: Learning Representations with CCA . . . . .	71
3.4	Application on Abstractive Document Summarisation . . . . .	75
3.4.1	Decoding Module . . . . .	75
3.4.2	Experimental Setup . . . . .	78
3.4.3	Discussion . . . . .	78
3.5	Conclusions . . . . .	79
<b>4</b>	<b>Canonical Correlation Inference for Image Captioning</b>	<b>81</b>
4.1	Image Captioning . . . . .	83
4.1.1	Abstract Scenes . . . . .	85
4.1.2	Evaluation . . . . .	86
4.2	Multi-view Representation Learning for Image Captioning . . . . .	90
4.2.1	Learning . . . . .	90
4.2.2	Decoding . . . . .	93
4.3	Experiments . . . . .	94
4.3.1	Experimental Setup . . . . .	95
4.3.2	Results . . . . .	97
4.3.3	Human Evaluation . . . . .	98
4.4	Conclusions . . . . .	102
<b>5</b>	<b>Multi-view Inference for Movie Understanding</b>	<b>103</b>
5.1	Movie Understanding . . . . .	105
5.2	Multi-view Sequential Inference . . . . .	107
5.3	Experiments . . . . .	112
5.3.1	Experimental Setup . . . . .	113
5.3.2	Perpetrator Mention Identification . . . . .	113
5.3.3	Episode Structure Tagging . . . . .	118
5.4	Conclusions . . . . .	120
<b>6</b>	<b>Narration Generation from Video Input</b>	<b>121</b>
6.1	The Peppa Pig Dataset . . . . .	124
6.1.1	Narration as Video Summary . . . . .	127
6.2	Narration Generation . . . . .	129
6.2.1	Narration Timing . . . . .	130
6.2.2	Content Generation . . . . .	131
6.3	Experiments . . . . .	132

6.3.1	Experimental Setup . . . . .	132
6.3.2	Experiments on Narration Timing . . . . .	134
6.3.3	Experiments on Content Generation . . . . .	134
6.4	Conclusions . . . . .	140
<b>7</b>	<b>Conclusions</b>	<b>141</b>
7.1	Future Work . . . . .	142
7.1.1	Applications of Multi-view Representation Learning . . . . .	142
7.1.2	Peppa Pig Dataset . . . . .	143
<b>A</b>	<b>Implementation Details</b>	<b>145</b>
A.1	Encoder model used in Deep CCA Extractive Summariser (DCorES + en-coder) . . . . .	145
A.2	Multi-view sequential inference model . . . . .	148
<b>B</b>	<b>Peppa Pig Dataset Details</b>	<b>149</b>
B.1	List of Episodes . . . . .	149
B.2	Preprocessing . . . . .	153
	<b>Bibliography</b>	<b>155</b>



# List of Figures

1.1	An example of data where a multi-view approach is intuitively meaningful.	4
1.2	A second example of data where a multi-view approach is intuitively meaningful. . . . .	5
1.3	t-SNE projection of multi-view representations, generated by different methods for a noisy version of the MNIST dataset. . . . .	7
2.1	Example of different representations of the same 2-dimensional data. . . . .	12
2.2	t-SNE projection of several multi-view representations generated by different methods for a noisy version of the MNIST dataset. . . . .	14
2.3	The three modes of view fusion: early, late and hybrid. . . . .	17
2.4	Overview of the architecture of Deep Canonical Correlation Analysis. . . . .	22
2.5	Overview of the architecture of Deep Canonical Correlation Analysis with Stochastic Decorrelation Loss. . . . .	24
2.6	Overview of the architecture of the Correlational Neural Network. . . . .	28
2.7	Overview of the architecture of Correlational Autoencoders. . . . .	29
2.8	Recurrent Neural Network. . . . .	30
2.9	Architecture of an LSTM cell. . . . .	32
2.10	Architecture of a GRU cell. . . . .	33
2.11	General unfolded overview of a multi-view sequence model. . . . .	34
2.12	Architecture of a multi-view LSTM (MV-LSTM) cell. . . . .	37
2.13	Architecture of the correlational GRU (corrGRU) cell. . . . .	41
3.1	An article from the CNN website with three summaries. . . . .	49
3.2	Overview of the proposed CorES and CorAS summarisation systems. . . . .	53
3.3	Overview of the learning module involved in multi-view document summarisation. . . . .	55
3.4	Schematic of the encoder-dcca model. . . . .	63

3.5	Learning with DeepCCA: choosing a threshold value for the dataset. . . .	67
4.1	An example photo from the MS-COCO dataset, along with annotations and captions. . . . .	82
4.2	Example photos from the VQA dataset and corresponding questions. . .	83
4.3	The set of objects and cliparts used in the Abstract Scenes Dataset. . . . .	85
4.4	An image with six descriptions (captions) from the Abstract Scenes dataset.	86
4.5	Demonstration of CCA inference. . . . .	92
4.6	Examples shown to annotators for image captioning human evaluation. .	100
4.7	Scatter plot of BLEU scores versus human ratings, for both CCA and SMT methods. . . . .	101
4.8	Examples of outputs from the SMT and CCA inference systems. . . . .	102
5.1	General unfolded overview of a multi-view sequence model. . . . .	108
5.2	Hierarchical multi-view recurrent model with multi-head attention. . . .	112
5.3	An excerpt of the CSI dataset, where the image and text modalities and annotations are shown. . . . .	114
6.1	An excerpt of the Peppa Pig dataset. . . . .	122
6.2	A snapshot from a news summary video which includes a form of narration (text shown between video excerpts to fill in gaps in the story). . . . .	124
6.3	Histograms of characteristics of Peppa Pig narrations. . . . .	126
6.4	Narrations and plot summaries from two episodes, <i>Episode 1: Muddy Puddles</i> and <i>Episode 7: Mummy Pig at Work</i> . . . . .	128
6.5	An excerpt of an episode of the Peppa Pig dataset, where all the levels of segmentation and tagging schemes for narration timing are shown. . . . .	131
6.6	Dialogue Video Narrator (DiViNa) and Dialogue Video Narrator with future dialogue encoding (Di <sup>2</sup> ViNa) models. . . . .	133
6.7	Multimodal Decoder used in <i>DiViNa+mmd</i> and <i>Di<sup>2</sup>ViNa+mmd</i> models and comparison with Text-only Decoder. . . . .	138
6.8	Examples of the output of Di <sup>2</sup> ViNa+mmd model, paired with respective ground truth narrations. . . . .	140
A.1	Schematic of the encoder-dcca model. . . . .	147

# List of Tables

3.1	Learning with DeepCCA: assessing different input setups and architectures	62
3.2	Learning with DeepCCA: comparing selection by classification and selection by representation. . . . .	65
3.3	Learning with DeepCCA: impact of different metrics on the performance of DCorES. . . . .	66
3.4	Learning with DeepCCA: projection dimensionality. . . . .	68
3.5	Learning with DCCA: ablation experiment considering more than two views.	71
3.6	Learning with DCCA: comparing DCorES with other summarisers. . . .	72
3.7	Learning with CCA: ablation for different inputs and selection setups in the DailyMail part of the dataset. . . . .	73
3.8	Learning with CCA: comparing CorES with other summarisers. . . . .	74
4.1	Example of phrases and their learnt probabilities. . . . .	96
4.2	Scene description evaluation results comparing the systems from Ortiz et al. to our CCA inference algorithm. . . . .	99
4.3	Average ranking by human judges for cases in which the caption has an average rank of 3 or higher and when its average rank is lower than 3, for both CCA and SMT. . . . .	101
5.1	Comparing unidirectional and bidirectional variants of early fusion models and our multi-view model. . . . .	115
5.2	Assessing the contribution of the components of our model. . . . .	116
5.3	Ablation experiment assessing the contribution of each modality in our multi-view model. . . . .	117
5.4	Comparing the performance of early fusion (EF) and multi-view (MV) models with attentive early fusion models. . . . .	118



5.5	Performance of early fusion (EF) and multi-view (MV) model variants on speaker type and case tagging. . . . .	119
6.1	Statistics of the Peppa Pig dataset. . . . .	125
6.2	Multimodal datasets for tasks related to text generation from videos. . . . .	127
6.3	Summarisation evaluation comparing the plot summaries to the corresponding narration sentences of each episode. . . . .	129
6.4	Results for narration timing. . . . .	135
6.5	Results for narration content generation. . . . .	136

# Chapter 1

## Introduction

The last decades have seen the growth of Natural Language Processing (NLP) to a vibrant research field. The ubiquity of text or speech data makes NLP research relevant and appealing to numerous applications, unfolding an unprecedented potential for interdisciplinary research. Consequently, ties have been strengthened between the NLP community and other communities, either related to artificial intelligence, such as Computer Vision (CV) or less related to computer science, such as social sciences and arts.

The fact that communities used to or still may work in isolation, is not entirely justified from their long-term goals or technological outcomes. A glowing example of an application spawn out of the convergence of speech and language technology communities is speech-enabled personal assistants, a technology that has been popularised with commercial products such as Siri, Alexa, Google Home and others. Instead of adopting a holistic view, such systems commonly follow a pipeline approach, wherein modules developed by each community are combined. A typical usage scenario is the following: a user's speech signal is transcribed by a speech-to-text module, language understanding modules operate on the generated text and provide the needed information for the response, while, finally, a speech synthesis module utters the responses to the user. Each module can be created separately, without having to interact with the others before the final deployment.

A pipeline paradigm allows for decoupled investigation of the different modules, but it does not come without limitations. Each of the modules is developed in an isolated environment, using module-specific objectives that may be artificial to some extent. This issue extends to performance metrics, too, since the final system is evaluated in a hugely different manner than each of the modules. In a related note, pipeline architectures suffer from error propa-

gation, that is errors in early stages being responsible for more serious than expected errors in subsequent modules. Finally, in a pipeline, each of the modules simplifies and normalises the input as needed, depriving the next modules of potentially useful information.

Data simplification and normalisation is considered standard practice, even in models that are not part of a pipeline. Data in the wild (Ang et al., 2013) come in all types, shapes and sizes, while datasets compiled for training machine learning models usually contain more information than what is used (or is absolutely necessary) for the task for which they were collated. For instance, news articles, a data domain of choice for several natural language processing applications, consist of more than just a body of text. Either in print or digital format, they may include a title, interim section headings or one or more accompanying images. Articles on web news portals may go further to employ videos, links to other articles, reader comments or relevant social media posts. Regardless of their types, all these extra information complement the knowledge distilled in the article. This is also the case for other, more diverse types of datasets that have started to gain interest by the NLP community lately. Electronic health records may consist of a mix of structured (such as prescriptions) and unstructured text (physician reports), images, videos or sound recordings from medical tests or demographic data (Birkhead et al., 2015). Another example is social media posts which can include text, images, videos, recordings, hyperlinks, or combinations thereof.

Typically, the application of machine learning techniques implies a simplification of datasets, performed as a preprocessing step. This way, data are massaged into a format appropriate and focused to the question that algorithms are developed to tackle. One of the first steps of such preprocessing for NLP datasets is stripping data of any formatting (stress in form of boldface or italics typeface, titles etc.) and any non-textual information. Recent works which have undertaken major simplification to convert modern news portal webpages to simple, easy to manipulate texts, are the CNN/Daily Mail (Hermann et al., 2015) and NewsQA (Trischler et al., 2017) datasets.

This type of simplification is of utmost importance in dissimilar and inconsistent datasets, since operating on complex and variable inputs may obscure the results and unnecessarily complicate the process. Focusing on simplified data, of one modality or one type of input isolates the problems to be solved. This strategic decision is not unique in machine learning or natural language processing; it is inherent in the scientific method. Another core reason why stripping datasets of particular kinds of information is necessary, is that doing so, makes up for a smooth approach to input representation, which is indispensable in machine learning applications. Machine learning algorithms operate on representations of input data

points in vector spaces, which can be sets of hand-crafted features, calculated features or learnt representations. The type of features or selected representation has proven to be of great significance over the course of machine learning history. Especially before the advent of distributed representations, when the selection of features was manually performed by domain experts or knowledge engineers, the opinion that “*features determine much of the success of a machine learning application*” (Flach, 2012) was prevalent. The importance of representations has led to the development of the field of representation learning, an active area of research, with popular approaches being constantly re-evaluated and new techniques proposed (Devlin et al., 2019; Peters et al., 2018).

Generating effective and informative input representations is a complex and challenging problem. The variability of data often brings out additional representational challenges, which can be directly tackled or circumvented by simplification. For instance, multimodal setups, where data come from more than one modalities, each of which can be represented using a large number of techniques, have to also face the additional burden of combining all those representations to one. Representation combination, commonly referred to as *representation fusion*, is an open research problem. Several approaches have been proposed and there has been no consensus on one, best all-around framework. Selecting a specific method for fusion, either arbitrarily or based on empirical data, amounts to an assumption for the problem at hand.

However, the more the assumptions about the input, the further the model is from real-world situations. It seems that humans do not find it particularly challenging to combine information coming from several modality sources: a reader of a news portal can effortlessly combine information from the text of an article, its accompanying images and videos. A trained physician is able to paint a good picture of the status of a patient (to the best of their ability) relying on information from all different formats of medical tests or prescriptions. Interestingly, there is evidence of modality integration rooted in human cognition; humans integrate audio-visual information in speech processing (“McGurk effect”). The McGurk effect (McGurk and MacDonald, 1976) describes, for example, the perception of a /*da*/ syllable by human subjects, when a visual /*ga*/ and a voiced /*ba*/ stimuli are provided at the same time.

The problem of fusion is not unique in multimodal setups, though. In purely unimodal settings, a fairly straightforward quality of input data that is often overlooked by both researchers and practitioners is the fact they they can be perceived, either physically or conceptually, as having multiple aspects or facets. Consider for example, the problem of represent-

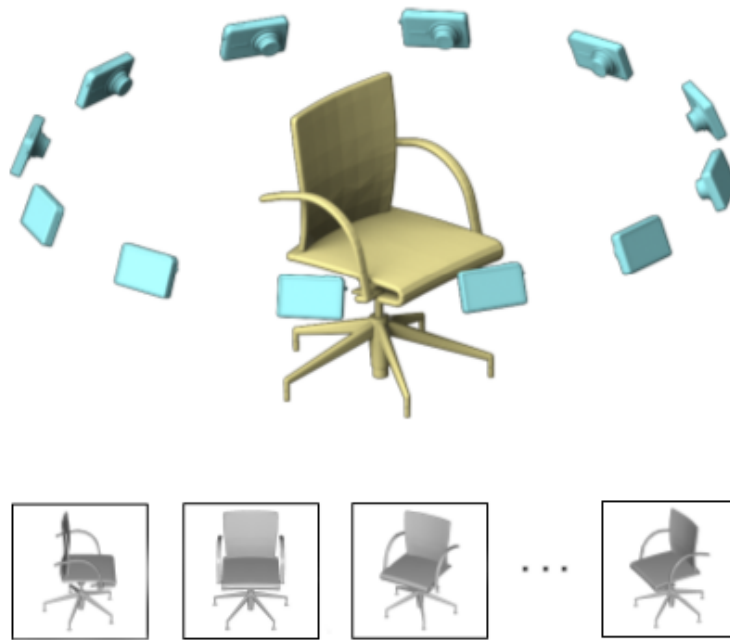


Figure 1.1: An example of data where a multi-view approach is intuitively meaningful. A 3D object (chair) is represented by a set of 2D images taken from different angles around it. Some photographs may include (or hide) information not present (or present) in others. Image adapted from the work of [Su et al. \(2015\)](#).

ing a simple three-dimensional (3D) object, such as the chair shown in Figure 1.1. Assuming that a modelling decision to represent 3D objects by using 2D photographs has been made, an efficient way to combine information from multiple 2D photographs is vital.

The need to take into account multiple viewpoints of the input is obvious in the example of the 3D chair: the viewpoint of the observer (photographer) plays an important role to the representation. Different angles can give vastly different perceived images, each of them including (or hiding) details that may not be present in others. Importantly, some of those angles may hide features of the object that are fundamental: for example, a plan (a view of the chair seen from above) does not reveal the fact that there is a surface designed for a person's back, the same way that a floor plan does not specify the height of a building's walls. Representing a 3D object may benefit from a more elaborate approach: considering more than one viewpoints (by taking photographs from different angles) and combining the photographs in one vector, that can represent the chair object. Following this approach, the initial representation problem is transformed to one that has to take into account multiple views of the input.

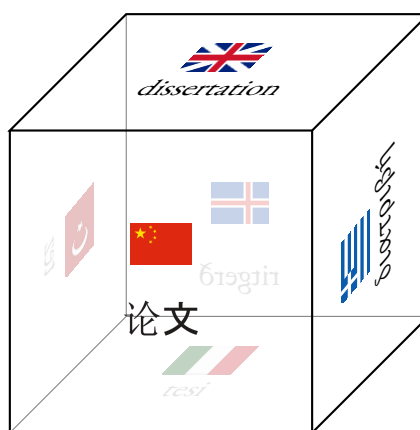


Figure 1.2: A second example of data where a multi-view approach is intuitively meaningful. Using information from different languages, coherent and informative word representations can be created.

A more representative example of multi-view nature can be found in multilingual NLP. Keeping in mind that the ultimate goal of generating word representations is to include the semantics of each word and sense in one vector, translations of words in different languages can be exploited. From an abstract point of view, translations of the same words in different languages can be thought of as different views of the word meaning, as shown in Figure 1.2. This is an example of multi-view setup that is not multimodal, since words in all languages belong to the same textual modality.

All multi-view setups, including the examples mentioned above, face the problem of representation fusion. The set of machine learning techniques that combine information from several input views to one, common representation is the object of *multi-view representation learning*. These techniques assume the presence and observation of two or more views of the data, which are fused into one. There are several benefits from adopting such an approach:

- The fact that one considers more than one views may enrich the information considered from the input space. For instance, in an emotion recognition task where the audio signal and the text transcript of a speech excerpt are given, using information from the audio can reveal sentiment cues, even if the content of the words does not imply it. For example, a person saying “I didn’t like this movie” can say it with an indifferent or angry voice; without considering the audio signal and simply focusing on the text, the emotion cannot be retrieved.
- Creating a common representation instead of simply aggregating representations from available views, results in reduced size of both the representations and the models op-

erating on them.

- The created representations may be devoid of redundant information. The fact that each view is (or is treated as) separate, does not imply that all information from it is unique. In our previous example of emotion recognition, the content of the speech is present in both the audio channel and text transcription of the signal. Combining the two views (text and audio) efficiently may reduce the amount of duplication in the final representation.
- Assuming that the available views are observed variables which have a common latent characteristic, the combined representation may be closer to the original, latent piece of information.
- In the process of creating shared representations, one can take into account cross-view interactions, that would go unnoticed otherwise. The combination of a phrase with seemingly neutral connotation (e.g. “It was OK, I guess.”) with audio cues implying frustration, may indicate disappointment.
- Combining information from available modalities is intuitive and can be justified as being close to human behaviour.

Multi-view approaches are justifiably fit or applicable in a wide range of problems and datasets; they can be used to model input from different poses for face recognition (Gross et al., 2008), from photos and sketches (Wang and Tang, 2008), colour and texture from images (Tzortzis and Likas, 2012), words and topics of documents (Gupta et al., 2019) or information from multiple sensors in time series classification (Xu et al., 2018).

Indeed, multi-view representation learning techniques have been extensively studied within computer vision problems, ranging from facial expression recognition (Zheng et al., 2006) to image clustering (Cai et al., 2013). They have also been researched in multimodal setups, such as cross-modal retrieval (Quadrianto and Lampert, 2011), audio-visual correspondence (Arandjelovic and Zisserman, 2017), audio-visual emotion recognition (Tian et al., 2001) and cross-view retrieval tasks (Holzenberger et al., 2019). Despite NLP research being mostly concerned with text-only problems, multimodal problems with a textual component, such as opinion mining (Somasundaran et al., 2006), subjectivity analysis (Raaijmakers et al., 2008) or sentiment analysis (Morency et al., 2011) have gained attention lately.

Research on *multimodal* and *multi-view* methods is, more often than not, parallel. Their difference is one of scope: multi-view algorithms can operate on data that can be seen by dif-

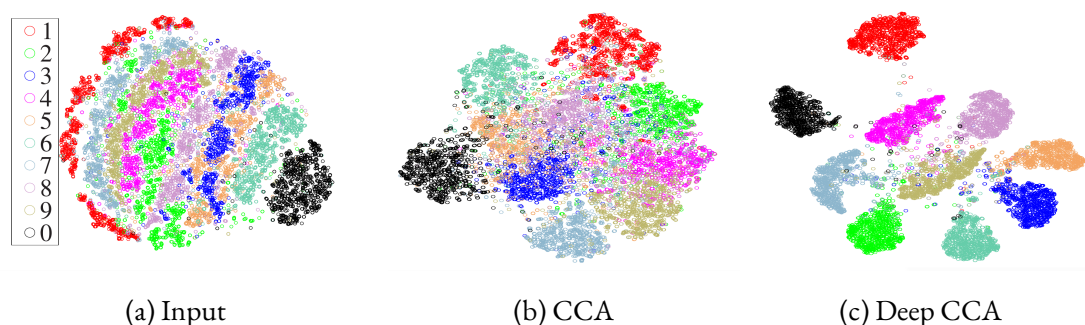


Figure 1.3: t-SNE projection of multi-view representations, generated by different methods for a noisy version of the MNIST dataset. The techniques shown are CCA (Hotelling, 1935) and Deep CCA (Andrew et al., 2013), both unsupervised. Each color refers to a different digit. Image from the work of Wang et al. (2015a).

ferent aspects (or views), while multimodal applications are mostly concerned with datasets that include different modalities (for example, image, audio and text). While the term *multi-view* does not always refer to multimodal settings, ideas from multi-view representation learning research are especially compelling for multimodal applications.

Despite evidence suggesting that the choice of representation plays an important role to downstream tasks, advances in representation learning are not always being brought into the spotlight. There are practical reasons for that, with the main argument being that simple heuristics give good enough performance, so that the use of complex representation methods is not justified. Even for fundamentally multimodal NLP problems, such as multimodal grounding or multimodal translation, multi-view representation learning techniques are not quite popular. Representation of multimodal inputs is commonly achieved by simple aggregation of representations of each modality. Examples of multi-view multimodal NLP-related work are limited to tagging short videos, such as the work of Zadeh et al. (2018a) on emotion recognition. Another reason why complex representation tools are not preferred is that intrinsic evaluation of representations is notoriously fuzzy. Although in some cases it can provide useful observations, as do, for example, the plots of Figure 1.3, most often, representation effectiveness is evaluated in an extrinsic manner, by gauging the empirical impact they have to a downstream task.

This thesis describes a multi-view contemplation of several NLP problems, both multimodal and non-multimodal in nature. The main argument put forward is that multi-view learning offers powerful representation learning tools that increase the performance of models on downstream tasks; hence they can and should be seriously considered by researchers and



practitioners of the NLP community. In short, we set out to explore the limits of the seeming applicability of multi-view representations to a range of tasks, from static, two-view, unimodal to dynamic, three-view, trimodal applications, rooted on structured prediction and text generation. In exploring the limits of multi-view representations, multimodality is approached from different angles. On the one hand, we set out to investigate whether multi-view representations can hold all the information required for complex multimodal problems, such as image captioning, while on the other hand, we explore whether multi-view representations are a good alternative to simple aggregation of representations. In most cases, carefully selecting a multi-view representation technique or employing a multi-view framework proves beneficial for the problem at hand.

## 1.1 Contributions

The present thesis investigates correlational multi-view learning techniques for Natural Language Processing applications. Specifically, the contributions made are the following:

- *multi-view summarisation*: we develop a multi-view framework for the problem of automatic document summarisation. The underlying idea is that, from an abstract point of view, a document and its summary constitute two views of the same latent semantics. We experiment with abstractive and extractive summarisation, demonstrating that correlational multi-view learning can be used for complex, downstream NLP applications that are not necessarily multimodal in nature.
- *multi-view image captioning*: we present a multi-view approach for image captioning. The core of this method is the construction of a joint multimodal space, describing both the images and captions. A novel algorithm makes use of this multimodal, multi-view space to infer text captions from images.
- *multi-view sequential inference for movies*: we develop a multi-view model for content-based video tagging. Our work places particular importance to the temporal nature of videos, which gives rise to a dual mode of analysis: incremental inference (models that mimic a human viewer watching a movie for the first time, thus having no knowledge about the future) and non-incremental inference (models that have look-ahead capabilities).
- taking a more abstract approach, we define *shallow movie understanding* as a set of problems revolving around content-based processing of movie and television series

videos. To this end, we introduce two novel tagging tasks for an existing movie dataset.

- we introduce *narration generation*, a new task for text generation from movie videos. We believe that the introduction of this task will challenge existing techniques for text generation from videos as it requires inference on the storyline of movies videos and temporal context-based reasoning. We collect a dataset for this task, and report results on several models on it, establishing a baseline and comparing it with multi-view models.

## 1.2 Overview

This thesis is organised as follows:

- *Chapter 2* introduces the concepts and necessary background in representation learning and multi-view learning and discusses several relevant approaches that have been proposed in the literature.
- The main premise of multi-view learning does not preclude its application to setups with non-multimodal data. In *Chapter 3*, we explore such a direction and present a multi-view framework for the problem of document summarisation.
- Multi-view techniques are particularly appealing for multimodal applications. *Chapter 4* outlines our work on image captioning. We propose a novel algorithm for generating sentences from a multimodal space, where both captions and images are projected. Experiments show that this approach can outperform previously proposed algorithms in generating captions for abstract images.
- *Chapter 5* discusses work on three tagging/segmentation tasks, that contribute to the broad agenda of movie understanding. For all three tasks, we propose a neural architecture paired with a novel training objective, designed for incremental inference over multi-view data. Thorough experiments confirm that multi-view representation learning is of crucial importance for movie-related tasks.
- In *Chapter 6*, we introduce *narration generation*, a novel text generation task from movie data. We formalise the task of narration generation, describe the process of collecting an appropriate dataset, and present a set of baselines and multi-view models on it.
- *Chapter 7* summarises the main findings of this thesis, discusses limitations and points

to future research directions.

### 1.3 Published Work

Parts of this thesis have been published as peer-reviewed papers. The work on Canonical Correlation inference and image captioning described in Chapter 4 was done in collaboration with Helen Jiang and Shay Cohen, who both carried out some of the experiments and contributed parts of a write-up that was later presented in the AAAI Conference on Artificial Intelligence (Papasarantopoulos et al., 2018). The work on multi-view movie understanding which is described in Chapter 5 was published in the Conference in Empirical Methods in Natural Language Processing (Papasarantopoulos et al., 2019). It was based on the dataset and implementation of [Fermann et al. \(2018\)](#).

# Chapter 2

## Background

*Features determine much of the success of a machine learning application, because a model is only as good as its features.*

*(Flach, 2012)*

A machine learning algorithm, from an abstract point of view, strives to find a relationship between an input space  $\mathcal{X}$  and an output space  $\mathcal{Y}$ . A huge amount of research work has been devoted in exploring ways to discover a function that maps from  $\mathcal{X}$  to  $\mathcal{Y}$  in various setups, contexts and datasets; the question of representation, though, that is how the space  $\mathcal{X}$  is defined, is equally important.

Commonly, machine learning algorithms operate on representations of input data points, where each data point is represented by a set of features. The selected features and the way they are encoded in the learning process are pivotal to the success or failure of machine learning systems. The importance of representation is also stressed by the fact that most problems include, in one way or another, some type of feature engineering, which, especially before the popularity of neural network techniques, could very frequently end up being labour-intensive, expensive, or calling for expert knowledge.

Specifically, it is generally admitted that the difficulty of a task varies with respect to the way information is presented (Goodfellow et al., 2016). The choice of a good representation depends not only on the task at hand or the model developed, but also on the data itself. Since representation evaluation can be fuzzy, more often than not, a good representation is considered one that makes the subsequent task easier to learn or solve, for a specific dataset. Even in simple setups, poor choice of representation can make the task difficult or impossible. A representative example is the task of building a simple linear classifier for the two-dimensional

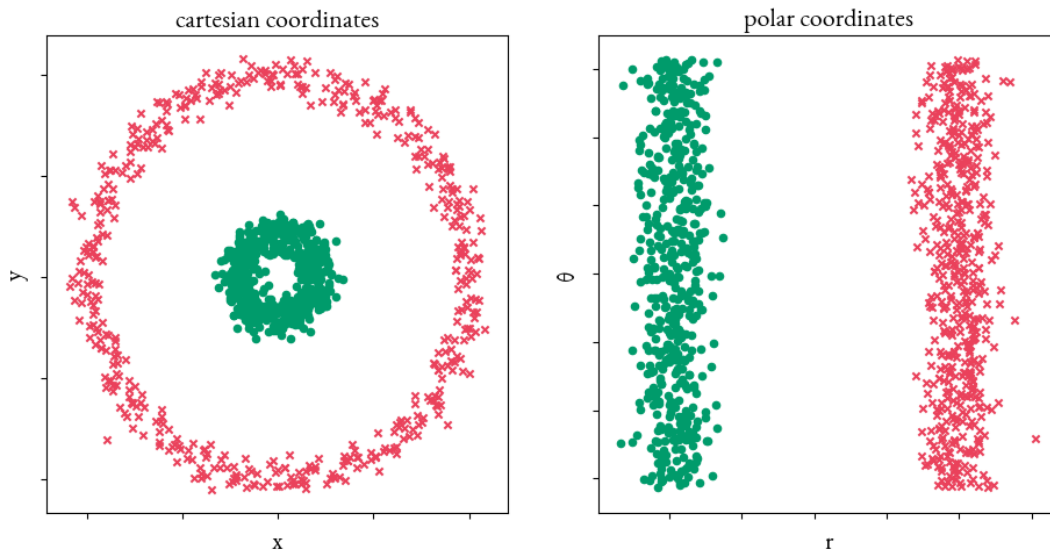


Figure 2.1: Example of different representations of the same 2-dimensional data. Evidently, considering the task of linear separation of the two clusters, representing data using polar coordinates makes the task not only possible (as opposed to cartesian coordinates), but also quite easy to solve.

data of Figure 2.1. The choice to express the input in polar coordinates (right) rather than cartesian (left) is crucial for the task at hand.

## 2.1 Representation Learning

*Representation learning* refers to the set of techniques devised to learn meaningful representations for the input of machine learning algorithms. Although in several fields and applications there has been a transposition of interest more to architecture engineering (Elsken et al., 2018), representation learning has grown to become a field itself with representation in leading community workshops and conferences, such as Neural Information Processing Systems (NeurIPS)<sup>1</sup>, International Conference on Machine Learning (ICML)<sup>2</sup> and International Conference on Learning Representations (ICLR)<sup>3</sup>.

Advances in representation learning for NLP have been propelled with the introduction of distributed representations (Hinton, 1986) and subsequent work in neural language modelling (Bengio, 2008), that resulted in techniques able to create diverse word representations,

<sup>1</sup><https://nips.cc/>

<sup>2</sup><https://icml.cc/>

<sup>3</sup><https://iclr.cc/>

which are now referred to as *word embeddings*. Distributed representations have been so successful, that, on a permanent basis, practitioners and researchers make use of pre-trained word embeddings<sup>4</sup> as a sole input representation for textual data.

Nowadays, there seems to be a huge variety of ways to create representations for any granularity of textual information, owing to the collective wisdom of the community that has experimented with various combinations of models and tasks. Combining distributed representations with neural network architectures has really pushed the limits of representation learning in NLP. Recurrent Neural Networks (RNN; Elman 1990), Convolutional Neural Networks (CNN; LeCun et al. 1998) and subsequent architectures, have paved the way for frameworks able to generate representations for words, phrases (Socher et al., 2013), sentences (Kim, 2014; Collobert et al., 2011) and whole documents (Le and Mikolov, 2014), starting from words or even characters.

Regardless of their core idea or the data they operate on, most representation learning problems “face a trade-off between preserving as much information about the input as possible and attaining nice properties of it” (Goodfellow et al., 2016). In unsupervised setups, attaining input properties can be an indicator of good quality of the representations. An example is shown in Figure 2.2, where different unsupervised techniques learn representations that make the subsequent classification task easier (as is the case with DCCA and DCCAE) as opposed to harder (as is the case with LLE).

The set of representation learning ideas that are referred to as multi-view learning capitalise on such a quality of the input data: the fact that it can be perceived, either physically or conceptually as having multiple aspects or facets. The plots of Figure 2.2 highlight this fact: they were generated in an unsupervised way by multi-view methods operating on digit images split in two parts (a left and a right).

## 2.2 Multi-view Learning

Machine learning algorithms usually treat all the characteristics of the training examples as features describing the input. The multi-view paradigm refers to settings where input data come from more than one, clearly distinct, source. The sources can be different modalities, as is the case with audiovisual or multimodal signals, or to the same modality, as is the case with parallel texts. Multi-view learning encompasses techniques designed to accommodate

---

<sup>4</sup>Popular choices include models such as *word2vec* (Mikolov et al., 2013a), *GloVe* (Pennington et al., 2014) or *ELMo* (Peters et al., 2018).

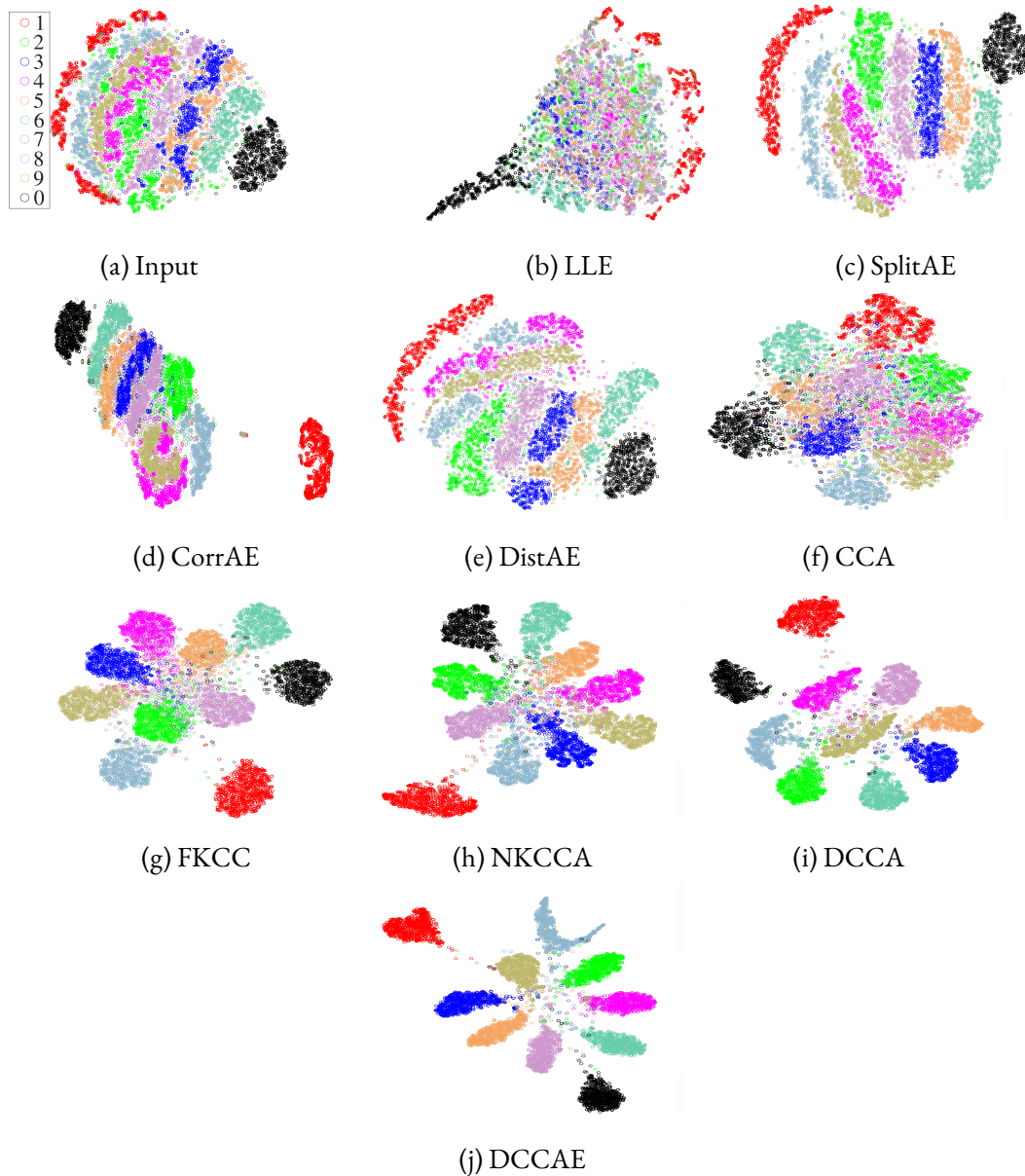


Figure 2.2: t-SNE projection of multi-view representations generated by different methods for a noisy version of the MNIST dataset. Each color refers to a different digit. The methods used are (b) Locally Linear Embedding (Roweis and Saul, 2000), (c) Split Autoencoders (Ngiam et al., 2011), (d) Correlational Autoencoders (Wang et al., 2015a), (e) Minimum Distance Autoencoders (Wang et al., 2015a), (f) Canonical Correlation Analysis (Hotelling, 1935), (g) Kernel CCA using random Fourier features (Lopez-Paz et al., 2014), (h) Kernel CCA using Nyström approximation (Williams and Seeger, 2001), (i) Deep CCA (Andrew et al., 2013), (j) Deep Canonically Correlated Autoencoders (Wang et al., 2015a). Neither of the feature algorithms use class information, but it is clear how different representations can help the task of classification to digits. Image from the work of Wang et al. (2015a).

more than one views in the learning setup. Such techniques are particularly attractive and interesting for intuitive reasons, especially for multimodal models, where modeling inter-modal (inter-view) interactions can be crucial to their predictions. Multi-view approaches are also attractive for theoretical reasons. [Anandkumar et al. \(2014\)](#) show that certain latent variable models, such as Markov Models, Gaussian Mixture Models and Latent Dirichlet Allocation Models can be optimally learnt with multi-view spectral algorithms.

Examining a large number of multi-view learning ideas, [Xu et al. \(2013\)](#) observe that there are two significant principles that underline them:

- the *consensus* principle states that the learning process aims to maximise the agreement on multiple distinct views. Supposing two independent hypotheses  $f^{(1)}$  and  $f^{(2)}$  on two different views  $X^{(1)}$  and  $X^{(2)}$  and their corresponding error rates  $P_{\text{err}}^{(1)}$ ,  $P_{\text{err}}^{(2)}$ , [Dasgupta et al. \(2002\)](#) show that, under the cotraining assumption ([Blum and Mitchell, 1998](#)), it holds that

$$P(f^{(1)} \neq f^{(2)}) \geq \max\{P_{\text{err}}^{(1)}, P_{\text{err}}^{(2)}\}, \quad (2.1)$$

where  $P(f^{(1)} \neq f^{(2)})$  is the probability of disagreement of the two hypotheses. This leads to the conclusion that minimising the disagreement rate of the two hypotheses will result in the error rate of each hypothesis to be minimised.

- the *complementarity* principle states that in a multi-view setting, each view of the data contains some knowledge that other views do not have. Different views should be able to contribute to the overall representation, meaning that they should clearly be different views and not just a random subset of the available features of the input.

The large number of multi-view techniques in the literature can be categorised as belonging to one of three groups ([Xu et al., 2013](#)):

- *co-training* ([Blum and Mitchell, 1998](#)) is a technique proposed for setups with small amount of labelled and large amounts of unlabeled data. Assuming two views (feature sets) of the data, a classifier is trained for each of the views on the labeled data and the most confident of the two is used to label some of the unlabelled data. Training is done alternately, ensuring the maximisation of the mutual agreement of the two views.
- *multiple kernel learning* is a technique to combine a predefined set of kernels. It was originally developed as a way to control the search space capacity of possible kernel matrices, but has been widely applied to problems involving data from multiple sources.
- *subspace learning* is the most widely used group of techniques and the group in which



the present thesis belongs. The core idea of such approaches is that there is a latent subspace from which the different views of data are generated. The objective of the learning process is to obtain such a shared subspace, commonly one with lower dimensionality than that of any of the spaces of the views.

*Multi-view representation learning* is almost exclusively concerned with subspace learning and it often surfaces as a tool for view fusion, that is a tool to create representations in this latent space, using the available multiple view representations. Generally, there are three alternatives when it comes to view fusion (Atrey et al., 2010):

- *early fusion* refers to a simple aggregation of representations at the feature level (commonly concatenation, but can also be summation, mean pooling or other). Early fusion can cause overfitting in the case of a small training sample (since the input would be described in a high dimensional space). Moreover, it is not intuitive, since representing each view separately may result in poor cross-view modelling. An abstract sketch of the function of early fusion can be seen in Figure 2.3(a).
- *late fusion* is the integration of outputs of different modality-specific modules. Instead of applying one model to the multi-view input, several models or components tailored to each of the available views are used. The output of those components is aggregated (by a simple aggregation function or a voting mechanism) to produce the final output. Figure 2.3(b) presents an abstract sketch of the function of late fusion.
- *hybrid fusion* refers to any technique that aims to extend early fusion, by creating sophisticated representations in which all available views are fused. For these approaches, representation learning is part of the overall machine learning model, as shown in Figure 2.3(c). This is the only group of techniques that involves actually *learning* representations instead of aggregating them. In this process, each view can act as regulariser, constraining possible representations, thus generalising better than one view. Usually, a big part of hybrid methods is devoted to *feature mapping*. In order to use representations from diverse vector spaces to describe the input, functions that map those spaces to a common feature space are needed. Feature mapping is not restricted to representation learning, though; creating a shared space to describe different inputs (or inputs and outputs) has been a cornerstone of retrieval problems. For example, in multimodal retrieval setups, where the goal is to retrieve images or videos corresponding to text snippets, projecting (mapping) text features and image (or video) features to a common space, makes retrieval possible. Finally, projection functions can also be

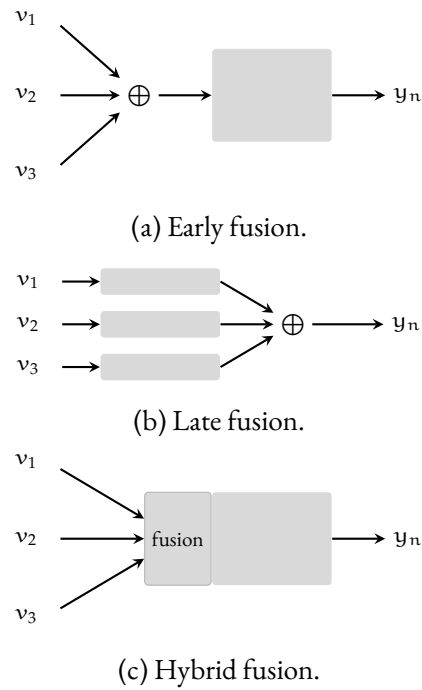


Figure 2.3: The three modes of view fusion: early, late and hybrid. In all figures, the grey boxes represent machine learning models. In early fusion, representations from the available views are aggregated before going through the model, while in late fusion there is a specific model for each of the views and the outputs of those models are aggregated. Hybrid fusion employs a middle ground approach, where fusion is part of the model.

useful in missing data setups; that is cases where in test time there is only a subset of the views available.

The present thesis, along with a plethora of works on the field of multi-view representation learning, attests that representations created by hybrid fusion methods are more likely to capture meaningful information and variation than those created with early or late fusion. The following sections enumerate and describe multi-view fusion techniques, following a practical division into *correlational* or *non-correlational*. Correlational techniques involve the calculation and/or maximisation of the correlation between views and are at large inspired by the classical statistical technique of Canonical Correlation Analysis (CCA; [Hotelling 1935](#)). Since this thesis clearly falls in this category, there is a heavier focus on relevant approaches.

### 2.2.1 Non-correlational approaches to multi-view learning

By *non-correlational*, we refer to techniques that do not involve the calculation of correlation between available views at any stage. From a severely abstract point of view, approaches that

perform early or late fusion can be also deemed as non-correlational. However, there are several ideas, mainly introduced for multimodal applications, that perform hybrid fusion without having a correlation orientation.

For example, the work of Zadeh et al. (2017) proposes a Tensor Fusion Network, wherein representations from three views (modalities) are fused in a way that accounts not only for trimodal, but also all possible combinations of bimodal and unimodal interactions. The work of Tsai et al. (2019) introduces a model that factorises representations into multimodal discriminative and modality-specific generative factors, by using a joint generative and discriminative objective, where the former part models modality interactions, while the latter accounts for noisy inputs and inferring missing modalities at test time. Finally, Multi-View Convolutional Neural Networks (MV-CNN; Su et al. 2015) are worth mentioning, which use view pooling to integrate information from more than one 2D images.

Earlier, non-correlational graphical models were designed for multi-view environments, with examples including multi-view sparse coding (Jia et al., 2010; Liu et al., 2014), multi-view latent space Markov networks (Chen et al., 2010) and multimodal Deep Boltzmann Machines (Srivastava and Salakhutdinov, 2012).

### 2.2.2 Correlational approaches to multi-view learning

*Correlational multi-view learning* refers to multi-view representation learning techniques that involve the calculation and maximisation of the correlation of available views in one way or another. The largest part of the work described in this thesis relies on such techniques.

#### 2.2.2.1 Canonical Correlation Analysis

Canonical correlation analysis (CCA; Hotelling, 1935) is a multivariate analysis technique, typically used in statistics to investigate commonalities between two variables. This is done by inferring linear combinations of the two variables that have maximum correlation with each other.

Considering two random vectors  $X \in \mathbb{R}^d$  and  $Y \in \mathbb{R}^{d'}$ , in a nutshell, CCA seeks projection functions  $u$  and  $v$  of the two variables to a shared space  $\mathbb{R}^m$  ( $u: \mathbb{R}^d \rightarrow \mathbb{R}^m$  and  $v: \mathbb{R}^{d'} \rightarrow \mathbb{R}^m$ ) that are maximally correlated. Specifically, CCA seeks projection matrices  $a \in \mathbb{R}^{m \times d}$  and  $b \in \mathbb{R}^{m \times d'}$ , so that the projections  $aX^\top$  and  $bY^\top$  are maximally correlated at each coordinate. Moreover, we assume that  $m < \min(d, d')$ , so that the projection is feasible and also acts as a dimensionality reduction process.

Formally, CCA solves the following optimisation problem:

$$\arg \max_{\mathbf{a}_j, \mathbf{b}_j} \text{corr}(\mathbf{a}_j \mathbf{X}^\top, \mathbf{b}_j \mathbf{Y}^\top), \quad (2.2)$$

where  $j \in \{1, 2, \dots, m\}$  are the coordinates of the projection matrices and  $\text{corr}$  is the Pearson correlation between the pairwise elements of the two vectors, which is defined as

$$\text{corr}(X, Y) = \frac{C_{XY}}{\sigma_X \sigma_Y}, \quad (2.3)$$

where  $C_{XY}$  is the covariance and  $\sigma_X$  and  $\sigma_Y$  the standard deviations of  $X$  and  $Y$  respectively.

The first pair of variables that maximise this quantity, if it exists, is called the *first pair of canonical variables*. Subsequent pairs of canonical (maximally correlated) variables can be calculated by making sure that they are uncorrelated with the already calculated pairs. Consequently, in order to find the  $k$ -th pair of canonical variables, one needs to ensure that Equation 2.2 holds, and at the same time

$$\begin{aligned} \text{corr}(\mathbf{a}_j \mathbf{X}^\top, \mathbf{a}_k \mathbf{X}^\top) &= 0, \quad k < j \\ \text{corr}(\mathbf{b}_j \mathbf{Y}^\top, \mathbf{b}_k \mathbf{Y}^\top) &= 0, \quad k < j. \end{aligned} \quad (2.4)$$

Finding canonical variables involves the calculation of the cross-covariance matrix  $C_{XY}$  of  $X$  and  $Y$ . Starting from Pearson correlation between the two variables  $X$  and  $Y$ , which is defined as

$$\text{corr}(X, Y) = \frac{C_{XY}}{\sigma_X \sigma_Y}, \quad (2.5)$$

and setting  $C_{XX}$  and  $C_{YY}$  the covariance of  $X$  and  $Y$  respectively, and  $C_{XY}$ , the correlation between the projections  $\mathbf{a} \mathbf{X}^\top$  and  $\mathbf{b} \mathbf{Y}^\top$  is defined as

$$\text{corr}(\mathbf{a} \mathbf{X}^\top, \mathbf{b} \mathbf{Y}^\top) = \frac{\mathbf{a}^\top C_{XY} \mathbf{b}}{\sqrt{\mathbf{a}^\top C_{XX} \mathbf{a}} \sqrt{\mathbf{b}^\top C_{YY} \mathbf{b}}}. \quad (2.6)$$

In practice, the empirical covariances are used to estimate  $C_{XY}$ ,  $C_{XX}$  and  $C_{YY}$ . Correlation is then defined as

$$\frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x}) \sum_{i=1}^n (y_i - \bar{y})}}, \quad (2.7)$$

where  $x_i$  and  $y_i, i \in \{1, 2, \dots, n\}$  are observations of  $X$  and  $Y$ , and  $\bar{x}$  and  $\bar{y}$  are mean values of  $x_i$  and  $y_i$  respectively.

Maximising the quantity of Equation 2.6 is equivalent to maximising the numerator with the constraint of the denominator being equal to 1. After a change of basis:

$$\begin{aligned}\Omega &= C_{XX}^{-1/2} C_{XY} C_{YY}^{-1/2} \\ c &= C_{XX}^{1/2} a \\ d &= C_{YY}^{1/2} b,\end{aligned}\tag{2.8}$$

Equation 2.2 is equivalent to

$$\arg \max_{c, d: \|c\|^2 = \|d\|^2 = 1} c^\top \Omega d.\tag{2.9}$$

The solution to Equation 2.9 can be computed by applying Singular Value Decomposition (SVD) on  $\Omega$ :

$$\Omega = C_{XX}^{-1/2} C_{XY} C_{YY}^{-1/2} \approx U \Sigma V^\top.\tag{2.10}$$

Often, only diagonal elements of those matrices are used, since the calculation of this version of CCA may contain the inversion of potentially large ( $d \times d, d' \times d'$ ) matrices. The outputs of CCA are projection functions  $u$  and  $v$ , for which the matrices  $a = C_{YY}^{-1/2} U$  and  $b = C_{XX}^{-1/2} V$  are returned.

CCA has been used as a representation learning tool in a variety of NLP problems, with applications in generating word embeddings (Dhillon et al., 2015; Osborne et al., 2016), multilingual applications (Haghighi et al., 2008; Faruqui and Dyer, 2014; Lu et al., 2015), semantic analysis (Vinokourov et al., 2002) and dimensionality reduction of multi-view text data (Rastogi et al., 2015). It is also an important sub-routine in the family of spectral algorithms for estimating structured models such as latent-variable PCFGs and HMMs (Cohen et al., 2012; Stratos et al., 2016) or finding word clusters (Stratos et al., 2014). Interestingly, CCA has also a probabilistic interpretation as a latent variable model for two Gaussian random vectors (Bach and Jordan, 2005).

While finding projection functions, CCA also accomplishes dimensionality reduction, since the dimensionality of the projected space is smaller than the smaller dimensionality of the two input spaces. In fact, the number of canonical correlations kept can be regarded as a

hyperparameter for learning problems. In Section 3.3.3.5, we measure the effect of dimensionality in the output of a summarisation system.

Despite its usefulness and elegance, CCA has two major limitations that may make it unfit for modern machine learning applications:

- it supports only two views, and there is not an obvious way to extend it to more.
- it can learn only linear mappings to the shared space.

Several CCA-inspired techniques have been proposed in the past to overcome those limitations:

- **Kernel CCA** (Hardoon et al., 2004) is an extension of CCA, which addresses the linearity problem by finding maximally correlated nonlinear projections, restricted to reproducing kernel Hilbert spaces with corresponding kernels.
- **Generalised CCA** (Horst, 1961) addresses the limitation on the number of views. It solves the optimisation problem of finding a shared representation for more than two views, by minimising the sum of the reconstruction error for each view. Generalised CCA does not only learn projections or transformations on each of the views, but also a view-independent representation that best reconstructs all of the view-specific representations simultaneously.
- **Neural models** that make use of CCA itself or correlation objectives that can overcome the limitation of linearity by passing each of the input views through stacked non-linear transformation layers. The following sections describe such architectures in detail.

#### 2.2.2.2 Deep Canonical Correlation Analysis

Deep Canonical Correlation Analysis (DCCA; Andrew et al., 2013) is a neural-network-inspired extension of the Canonical Correlation Analysis method. Following the success of neural network-based methods in representation learning, DCCA operates in two stages: it computes representations of two views by passing them through multiple stacked layers of non-linear transformations, which in turn are given as input to Canonical Correlation Analysis. The whole neural architecture is trained by maximising the correlation between the outputs of the layers of non-linear transformations.

The operation of the DCCA network is outlined in Figure 2.4. From a blackbox point of view, it learns projection functions  $u$  and  $v$  from the input spaces  $\mathcal{X}$  and  $\mathcal{Y}$  to a shared space,

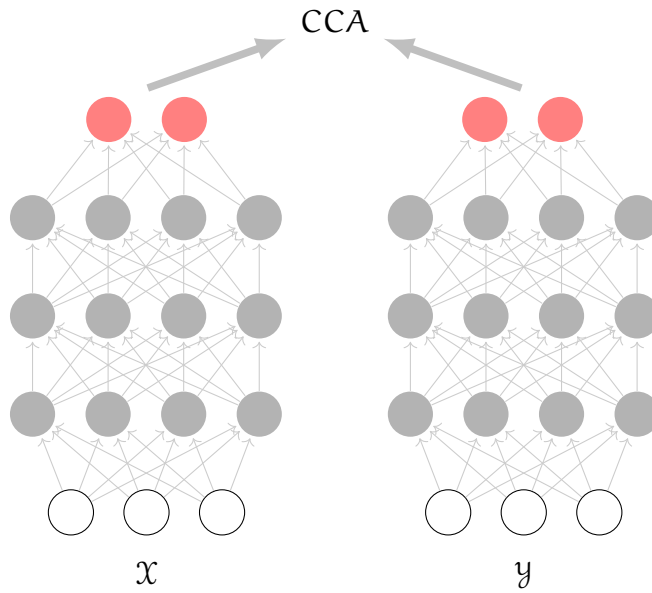


Figure 2.4: Overview of the architecture of Deep Canonical Correlation Analysis.

just like CCA. What makes DDCA distinct is that  $\mathbf{u}$  and  $\mathbf{v}$  are realised as composite functions:  $\mathbf{u} = \mathbf{u}_1(f_1(\mathbf{x}))$  and  $\mathbf{v} = \mathbf{v}_1(f_2(\mathbf{y}))$ , where  $f_1, f_2$  are the transformations learnt by the neural network and  $\mathbf{u}_1$  and  $\mathbf{v}_1$  the projections learnt by CCA. While the CCA layer operates exactly by the algorithm described in the previous section, the rest of the network needs an objective to be trained. The role of the objective is fulfilled by the correlation between the projected vectors in the CCA layer.

Formally, DCCA solves the following optimisation problem for two sets of parameters  $\theta_1$  and  $\theta_2$ :

$$\arg \max_{\theta_1, \theta_2} \text{corr}(f_1(\mathbf{x}; \theta_1), f_2(\mathbf{y}; \theta_2)), \quad (2.11)$$

Setting  $H_1 = f_1(\mathbf{x})$  and  $H_2 = f_2(\mathbf{y})$ , the total correlation of the top  $k$  components is the sum of the  $k$  singular values of the matrix  $\Omega = C_{XX}^{-1/2} C_{XY} C_{YY}^{-1/2}$  calculated for  $H_1$  and  $H_2$ . This is exactly the matrix trace norm of  $\Omega$ , or

$$\text{corr}(H_1, H_2) = \|\Omega\|_{\text{tr}} = \text{tr}(\Omega' \Omega)^{1/2}. \quad (2.12)$$

Training the network to optimise for correlation is not straightforward, since backpropagation would need the calculation of its gradient. To compute the gradient of the correlation

with respect to the parameters of the network, one can compute its gradient with respect to  $H_1$  and  $H_2$  and then use backpropagation. Then, the gradient with respect to  $H_1$  is calculated as follows:

$$\frac{\partial \text{corr}(H_1, H_2)}{\partial H_1} = \frac{1}{n-1} (2\nabla_{11}\bar{H}_1 + \nabla_{12}\bar{H}_2), \quad (2.13)$$

where  $\bar{H}$  stands for centered data matrices ( $\bar{H} = H - \frac{1}{n}H\mathbf{1}$ ). Using singular value decomposition ( $\Omega \approx U\Sigma V^\top$ ), [Andrew et al. \(2013\)](#) derive

$$\nabla_{12} = C_{11}^{-1/2}UV^\top C_{22}^{-1/2} \quad (2.14)$$

$$\nabla_{11} = -\frac{1}{2}C_{11}^{-1/2}U\Sigma U^\top C_{11}^{-1/2}. \quad (2.15)$$

Similarly,  $\frac{\partial \text{corr}(H_1, H_2)}{\partial H_2}$  has a symmetric expression.

The calculation of correlation is of pivotal importance to the operation of DCCA, since the optimisation process is guided by its value. Since the correlation is essentially a function of the entire training set and does not decompose into a sum over data points, there is no straightforward way to apply Stochastic Gradient Descent. This is the reason why the original proposal of [Andrew et al. \(2013\)](#) was that the model should be trained by full-batch optimisation using the L-BFGS second-order method. The intuition is that mini-batch techniques would fail to calculate correlations adequately. Moreover, the authors back up this intuition by mentioning conducting experiments with mini-batches of various sizes, failing to obtain satisfactory results.

The experiments conducted in support of Deep CCA make use of the MNIST dataset ([LeCun et al., 1998](#)), where images are split vertically, providing two views for digit classification. Moreover, the authors experiment with articulatory speech classification using the Wisconsin X-Ray Microbeam Database (XRMB; [Westbury 1994](#)) of simultaneous acoustic and articulatory recordings. The experiments show that Deep CCA outperforms CCA and Kernel CCA on both tasks.

While the DCCA algorithm brilliantly overcomes the linearity limitations of CCA, it does so at the expense of high computational cost, which derives from the full-batch optimisation constraint. Such a training paradigm, realistically, allows for DCCA to be applied only to small datasets, such as the MNIST or the XRMB. On the other hand, most modern neural



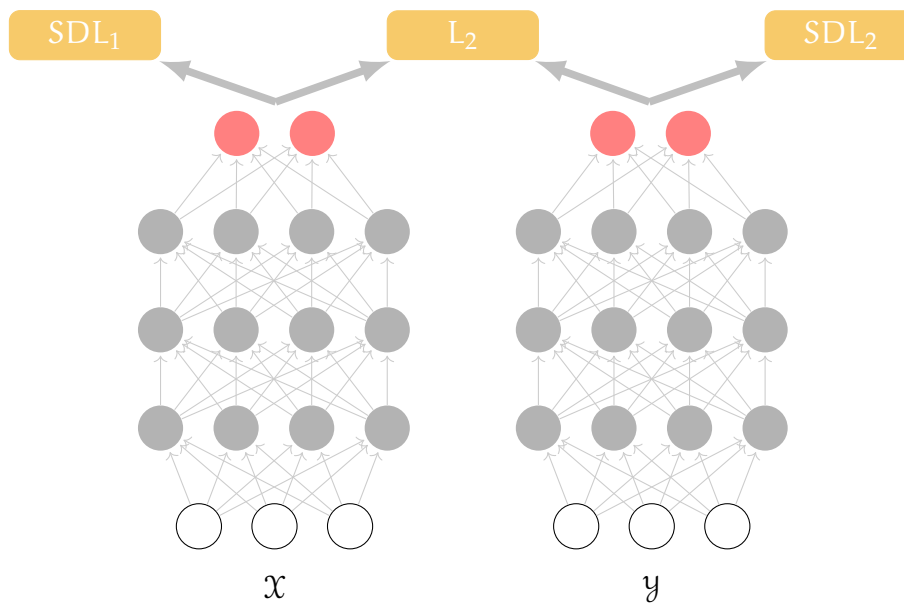


Figure 2.5: Overview of the architecture of Deep Canonical Correlation Analysis with Stochastic Decorrelation Loss.

network techniques use stochastic optimisation, enjoying fast training capabilities courtesy of hardware advancements related to Graphics Processing Units (GPUs).

### 2.2.2.3 Deep Canonical Correlation Analysis with Stochastic Decorrelation Loss

In the DCCA formulation of [Andrew et al. \(2013\)](#) and subsequent deep CCA works ([Wang et al., 2015b](#); [Yan and Mikolajczyk, 2015](#); [Wang et al., 2015c](#)), the representations created by the neural network  $f_1(x)$  and  $f_2(y)$  are decorrelated by forcing their correlation matrices over the training batch to be identity matrices. In order to do that, inversion of large matrices is required in each training step, which can be expensive.

The work of [Chang et al. \(2018\)](#) presents an alternative approach that allows for efficient computation, by relaxing the correlation objective. The proposed architecture is quite similar to the one of DCCA: two different networks encode the two views, and their output is used to calculate the loss and backpropagate the errors. Instead of relying to a full-batch covariance matrix  $C_{XY}$ , an incremental approximation of it is calculated, by collecting statistics from every mini-batch. Subsequently, instead of forcing a hard-decorrelation (forcing the off-diagonal elements of  $C_{XY}$  to be zero), decorrelation of each view is part of the model loss and training proceeds with the aim of minimising it along with any other losses of the model.

More specifically, in this approach, the full-batch covariance matrix is incrementally approx-

imated at each timestep, using an accumulative covariance matrix  $C_{\text{accu}}$  and a normalising factor  $c$ , as

$$C_{\text{apprx}}^{(t)} = \frac{C_{\text{accu}}^{(t)}}{c^{(t)}}. \quad (2.16)$$

The accumulative covariance matrix is updated in each time step as follows:

$$C_{\text{accu}}^{(t)} = \alpha C_{\text{accu}}^{(t-1)} + C_{\text{mini}}^{(t)}, \quad (2.17)$$

where  $C_{\text{mini}}^{(t)}$  is the covariance matrix of the mini-batch the model consults at timestep  $t$  and  $\alpha \in (0, 1]$  is a decay rate acting as a hyperparameter. This incremental stochastic learning setup is initialised with an all-zero matrix  $C_{\text{accu}}^{(0)}$ .

Instead of enforcing exact decorrelation (forcing the off-diagonal elements of  $C_{\text{apprx}}$  to be zero), the authors propose calculating a l1 loss on them:

$$\mathcal{L}_{\text{SDL}} = \sum_{i=1}^k \sum_{j \neq i}^k |\phi_{ij}^{(t)}|, \quad (2.18)$$

where  $\phi_{ij}^{(t)}$  is the  $(i, j)$  element of  $C_{\text{apprx}}^{(t)}$ . This loss is referred to as Stochastic Decorrelation Loss (SDL).

The gradient of this soft decorrelation loss can be calculated as

$$\frac{\partial \mathcal{L}_{\text{SDL}}}{\partial Z} = \frac{1}{c^t} \frac{1}{n-1} Z \cdot S, \quad (2.19)$$

where  $Z$  is the current mini-batch and  $S$  is a sign matrix

$$S(i, j) = \begin{cases} 1, & \phi_{ij}^{(t)} > 0 \\ 0, & i = j \text{ or } \phi_{ij}^{(t)} = 0 \\ -1, & \phi_{ij}^{(t)} < 0. \end{cases} \quad (2.20)$$

The idea of stochastic decorrelation loss was initially proposed as a general framework to accommodate various models, but one obvious testbed was the implementation of DCCA.

An overview of DCCA implementation with stochastic decorrelation loss can be found in Figure 2.5. In this figure, three loss terms are calculated: the decorrelation loss of each view and the l2 loss between the projected outputs. The use of the l2 loss is justified by the intuition that while decorrelating, a deep CCA model would like to reduce the distance between the projections of the two views at the same time.

The described formulation of the problem overcomes scalability issues that come up with the need to train in one large batch, as is the case with DCCA or other approaches that need to calculate a full CCA step for every (large-sized) mini-batch (Wang et al., 2015b). Moreover, the simple form of the loss makes this approach ideal for exploiting various deep learning frameworks, without having to result to complex coding and configuration issues. Using the stochastic decorrelation loss, one is able to enjoy the representational powers of the deep CCA paradigm using large datasets, while minimising scalability issues. Interestingly, Chang et al. (2018) demonstrate that a deep CCA model with that loss is able to reach a higher correlation strength than a DCCA model in cross-view digit recognition task on the MNIST dataset. They also report higher correlation than DCCA in a experiment with Multi-PIE (Gross et al., 2008), a substantially larger than MNIST dataset containing images of faces of people. We make use of the DCCA with Stochastic Decorrelation Loss in experiments with document summarisation (for details, see Chapter 3).

#### 2.2.2.4 Deep Generalised Canonical Correlation Analysis

As outlined in Section 2.2.2.1, one of the major limitations of Canonical Correlation Analysis is its inability to accommodate setups with more than two views. Generalised Canonical Correlation Analysis (GCCA) is an extension of CCA that fills this void, albeit calculating only linear transformations of the input. Inspired both by Deep CCA and GCCA, the technique of Deep Generalised Canonical Correlation Analysis (DGCCA; Benton et al. 2019) accommodates setups with more than two views while using non-linear transformations.

DGCCA passes representations of each view through multiple layers of non-linear transformations, in a manner analogous to DCCA. At the same time, it follows the objective of GCCA: it strives to find a shared representation of the different views, by not only learning a projection for each view, but also a view-independent representation that best reconstructs all of the view-specific representations simultaneously.

The authors derive a calculation for the gradient of the GCCA objective and learn representations using Stochastic Gradient Descent. They perform experiments on a restricted synthetic dataset and on the XRMB dataset, where they empirically show that DGCCA out-

performs DCCA (using acoustic and articulatory data) and GCCA (using the former two views and also phoneme labels as a third view). Moreover, they show that DGCCA is able to outperform PCA (Pearson, 1901), GCCA and a discriminative view weighting GCCA variant on a hashtag recommendation task (Benton et al., 2016).

### 2.2.2.5 Correlational Neural Networks

Chandar et al. (2016) design a multi-view representation learning neural network with three main features: any single view must be able to be reconstructed from the common representation, a view must be able to be predicted from the representations of other views and representations must be correlated. Their proposed model, *Correlational Neural Network* (CorrNN) has the overall structure of an autoencoder for two views.

More specifically, considering data  $z = (x, y)$  consisting of two views  $x \in \mathcal{X}$  and  $y \in \mathcal{Y}$ , the model learns a projection from  $\mathcal{X}$  and  $\mathcal{Y}$  respectively to the shared space:

$$h(z) = f(Wx + Vy + b), \quad (2.21)$$

with  $W, V$  and  $b$  learnable parameters and  $f$  any activation function. Moreover, it attempts to reconstruct the original views from the shared representation, generating data in spaces  $\mathcal{X}'$  and  $\mathcal{Y}'$ :

$$z' = g(W'h(z) + V'h(z) + b'), \quad (2.22)$$

with  $W', V'$  and  $b'$  also learnable parameters and  $g$  any activation function. The architecture of the Correlational Neural Network can be seen in Figure 2.6.

The CorrNN uses the following objective:

$$\mathcal{L} = \sum_{i=1}^L (\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{rec}}^{(X)} + \mathcal{L}_{\text{rec}}^{(Y)}) - \lambda \text{corr}(h(X, 0), h(0, Y)), \quad (2.23)$$

where  $h(X, 0), h(0, Y)$  refer to projected representations based on a single view,  $\mathcal{L}_{\text{rec}}$  stands for a self-reconstruction error (minimising the error in reconstructing  $x_i$  from  $x_i$ ),  $\mathcal{L}_{\text{rec}}^{(X)}$  is the reconstruction error from the first view (reconstructing  $y_i$  from  $x_i$ ),  $\mathcal{L}_{\text{rec}}^{(Y)}$  the reconstruction error from the second view (reconstructing  $x_i$  from  $y_i$ ),  $\text{corr}$  is the correlation between the projections of the two views and  $\lambda$  is a hyperparameter weighing the contribution of  $\text{corr}$ .

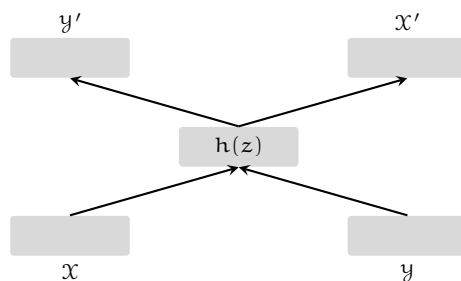


Figure 2.6: Overview of the architecture of the Correlational Neural Network.

Correlational Neural Networks have been tested in the task of digit classification with vertically split images, where they achieve higher correlation than DCCA. Interestingly, the authors do not engage in the issue of calculating correlations within a minibatch instead of the whole dataset, or any related scalability issues. They decided to compute the correlation term separately for each mini-batch.

### 2.2.2.6 Correlational Autoencoders

Correlational autoencoders make up a large section of correlational models. These models adopt the functionality of autoencoders, where the objective is to learn a compact representation that best reconstructs inputs. In order to fit to the multi-view paradigm, these models take two views as inputs and learn to reconstruct both of them, in a fashion similar to Split Autoencoders (Ngiam et al., 2011). Additionally, they use a correlation loss term to enforce correlation between the views.

An overview of the architecture of correlational autoencoders can be seen in Figure 2.7. Examples of such models are the Deep Canonically Correlated Autoencoder (DCCAE), the Correlated Autoencoder (CorrAE) and Minimum Distance Autoencoders (DistAE), all presented in the work of Wang et al. (2015a). These models use the same framework, but slightly different objectives, each with different levels of relaxation to decorrelation constraints. The authors highlight the effectiveness of the proposed models by experimenting with the MNIST and XRMB datasets, as well as with learning word embeddings from multilingual data.

It is clear from the description of the correlational multi-view approaches mentioned in the previous sections that multi-view representation learning ideas are commonly tested on synthetic datasets or carefully restricted benchmark setups. Within NLP, the role of such a benchmark is often assumed by the multimodal sentiment analysis problem. While careful evaluation on established datasets is really important to highlight their differences and

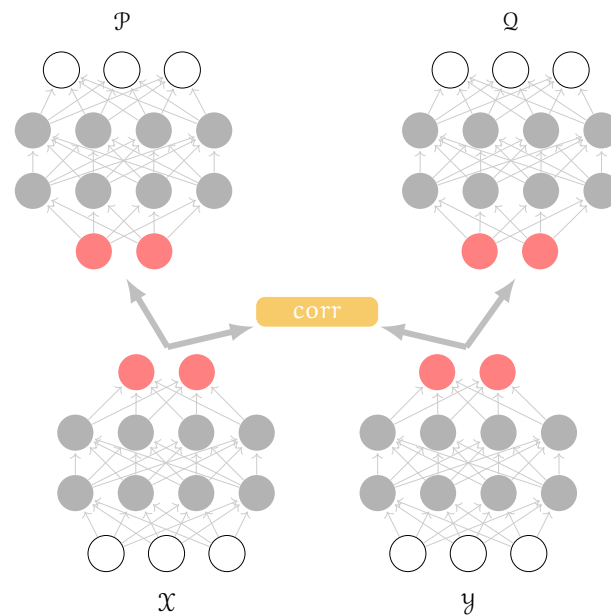


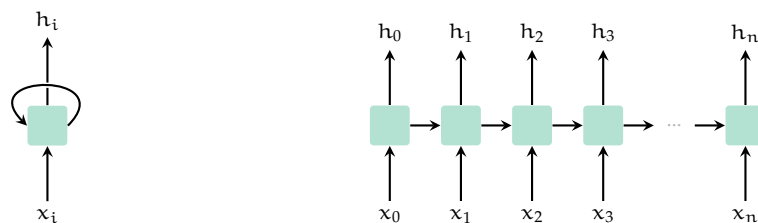
Figure 2.7: Overview of the architecture of Correlational Autoencoders.

their performance, in this thesis, we experiment with multi-view representation learning techniques in more complex, downstream, NLP applications.

## 2.3 Multi-view sequence learning

The approaches described in the previous sections mostly accommodate monolithic data points, and are not directly applicable to sequences. An abundance of practical applications rely on the efficient processing of sequential data, such as time series, temporal sequences (e.g. videos) or DNA sequences. The very nature of sequential data calls for particular modeling approaches. The main reason for that is that the data points are not independent and identically distributed (i.i.d.), but each is dependent on the previous (in the sequence) data points.

Natural Language Processing is interwoven with the use of sequence models, partially due to the view of language as a sequence of elements (phonemes, syllables, words, sentences). Sequential models, in one form or another, have been used in both speech and language applications. Following the popularity of Hidden Markov Models (HMM; [Baum and Petrie 1966](#)) and Conditional Random Fields (CRF; [Lafferty et al. 2001](#)) in language applications, neural sequence models have proven invaluable in NLP tasks, such as language modelling. They enjoy popularity as stand-alone models (as is the case with generative models), or as parts of more complex architectures (they are often a model of choice for sentence encoders).



(a) A Recurrent Neural Network cell. (b) An unrolled Recurrent Neural Network.

Figure 2.8: Recurrent Neural Network.

### 2.3.1 Sequence Models

Neural sequence models, especially before the advent of Transformer models (Vaswani et al., 2017), often follow the framework of Recurrent Neural Networks (RNN; Elman 1990; Medsker and Jain 1999). An RNN is a neural network with a loop that allows for information to flow along a temporal sequence. Such a network iterates over elements of sequences, maintaining a state vector about the sequence and emitting an output vector at each time step. Figure 2.8(a) shows an RNN cell, that is the basic structural unit that is repeatedly applied to the elements of sequences. Figure 2.8(b) shows the same network, unrolled over time, so that the flow of information is clearer. At each timestep, the model takes as input the current sequence element  $x_i$  and the previous internal state of itself. Subsequently, it produces an output  $h_i$  and updates its internal state, which is in turn used in the next step.

Depending on the context, an RNN can be used either as a discriminative or a generative model. Applications range from sequence labelling to sequence generation, or, simply encoding. In the case of sequence labelling, the emitted vector is used to infer a label for each of the elements of the sequence. Similarly, for generation, the emitted vector is used to infer a symbol. In the case of encoding, usually, the emitted vectors are not used; the RNN is rather used to process and create a single representation for a full sequence. Commonly, the final hidden representation of the model is used as the sequence embedding.

The element-by-element (step-by-step) processing has the intuitive implication of dividing the sequence to three parts: the *past*, the *present* and the *future*. In the generative paradigm, information from the past and the present is used to generate a prediction that will serve as the future. This is exactly the question of language modeling: after going through a sequence of tokens, a token that should follow those tokens is predicted. This process can be repeated multiple times to generate a full sentence or more than one sentences.

In the case of encoding, the sequence model is entrusted with the task of creating a repre-

sentation for the whole sequence or parts thereof. As such, and subject to data availability, it may have access to the full sequence and be able to use information from the past, the present and the future at each time step. A straightforward way to accomplish that is by using bidirectional networks (Schuster and Paliwal, 1997), that is, at each timestep, combining representations created by two separate sequence models: one processing the sequence from the beginning to the end and one operating the opposite way.

While the main idea of RNNs is elegant, it turns out that training simple RNNs is quite difficult, since they are prone to the exploding/vanishing gradient problem and are not particularly able to account for long term dependencies (Bengio et al., 1994); disqualifying shortcomings for language applications. To this end, gated models specifically designed to address these issues have been proposed. Perhaps the most popular gated model is the Long Short Term Memory (LSTM; Hochreiter and Schmidhuber 1997).

Broadly, an LSTM uses the RNN framework: it maintains a representation (state) for the whole sequence and as it iterates through the sequence elements, the state gets updated accordingly with the new information that comes to light. The strong point of the LSTM is that it relies on the use of gates that control the amount of information getting in the state, thus allowing for flexibility and control to forget parts deemed unimportant or stress important parts. The LSTM consists of a group of calculations that implement three gates: *input*, *forget* and *output* gate. The role of the forget gate is to calculate how much of the previous state should be forgotten, gauging the amount of information collected about the sequence so far that should remain in the state. The input gate controls what information carried by the current element should remain in the cell's state, while the output gate decides what the model will output.

More specifically, an LSTM maintains a cell state  $c_t$  and a hidden state  $h_t$ . The forget and output gates are defined by the following equations:

$$\begin{aligned} f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\ i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i), \end{aligned} \tag{2.24}$$

where  $\sigma_g$  is an activation function,  $h_{t-1}$  the hidden state of the previous step  $W_f, W_i, U_f, U_i$  weights and  $b_f$  and  $b_i$  are biases. Clearly, the outputs of both gates are based on both the input and the model's hidden state.

After visiting the current element, the LSTM updates its state, by combining information



from the forget and input gates:

$$\begin{aligned}
 \tilde{c}_t &= \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\
 c_t &= f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \\
 o_t &= \sigma_o(W_o x_t + U_o h_{t-1} + b_o) \\
 h_t &= o_t \circ \sigma_h(c_t).
 \end{aligned}
 \tag{2.25}$$

In the above equations  $\circ$  is the Hadamard (element-wise) product and  $\sigma_g, \sigma_h, \sigma_o$  are activation functions. Commonly, the sigmoid function is used as  $\sigma_g$ , while the hyperbolic tangent ( $\tanh$ ) is used for  $\sigma_c$  and  $\sigma_h$ . A schematic depiction of the LSTM cell can be seen in Figure 2.9.

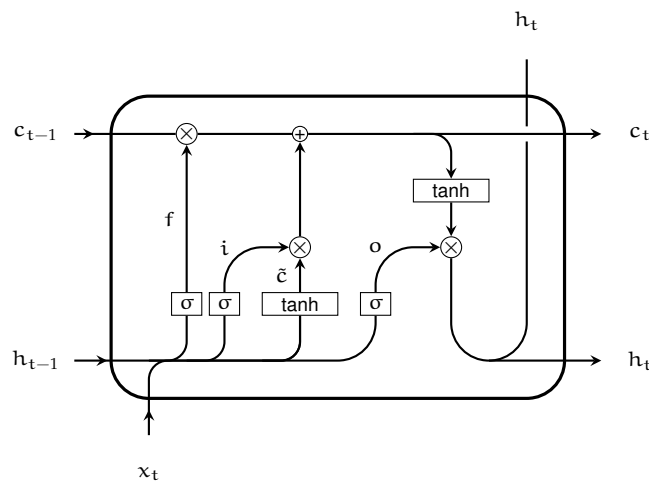


Figure 2.9: Architecture of an LSTM cell.

Several modifications of the LSTM have been proposed in the literature, in the quest of refining or simplifying the core gate calculations. Variants may use other activation functions (Gers and Schmidhuber, 2001), peepholes (Gers and Schmidhuber, 2000), or modified architectures (Krause et al., 2017). Empirical comparisons between LSTM variants, such as the works of Greff et al. (2016) and Jozefowicz et al. (2015), do not conclusively find significant differences between them.

Perhaps the most notable variant of the LSTM is the Gated Recurrent Unit (GRU; Cho et al. 2014). The architecture of a GRU cell can be seen in Figure 2.10. Its main differences from the LSTM are that it drops the forget and input gates and uses one *update* gate.

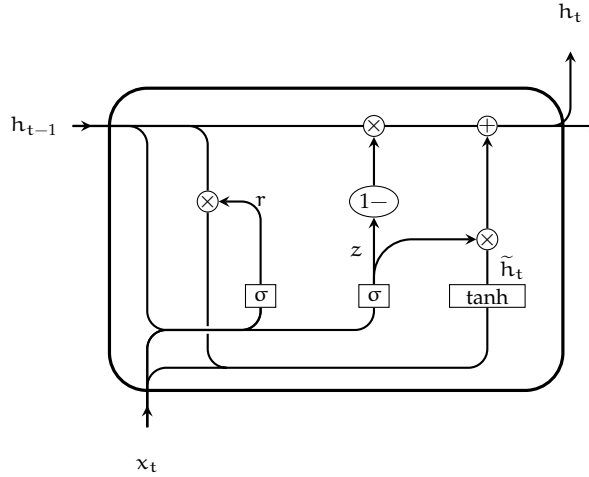


Figure 2.10: Architecture of a GRU cell.

Formally, the defining equations of the GRU are:

$$\begin{aligned}
 z_t &= \sigma_g(W_z x_t + U_z h_{t-1} + b_z) \\
 r_t &= \sigma_g(W_r x_t + U_r h_{t-1} + b_r) \\
 \tilde{h}_t &= \sigma_h(W_h x_t + U_h (r_t \circ h_{t-1}) + b_h) \\
 h_t &= (1 - z_t) \circ h_{t-1} + z_t \circ \tilde{h}_t.
 \end{aligned} \tag{2.26}$$

Similarly to the LSTM,  $\sigma_g, \sigma_h$  are activation functions (sigmoid and hyperbolic tangent respectively) and  $\circ$  is the Hadamard product.

The GRU was proposed as an alternative to the LSTM for an encoder-decoder architecture, its main asset being that it could replace the latter while having less parameters to learn. Although the GRU is reported to have some limitations (Weiss et al., 2018) and there is some evidence that in LSTMs perform better in encoders used for machine translation (Britz et al., 2017), there is also empirical work that reports no substantial differences between the two (Chung et al., 2014). While the issue of superiority seems to be application and dataset specific, especially in practical applications where training time is important, GRUs and LSTMs may be used interchangeably.

### 2.3.2 Multi-view Sequence Models

The aforementioned sequence models accept a single input from the sequence at each time. Given this restriction, their applicability to multi-view or multimodal setups boils down to

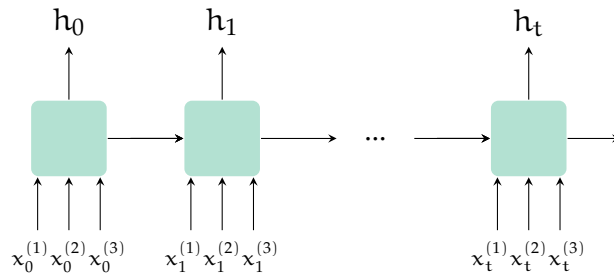


Figure 2.II: General unfolded overview of a multi-view sequence model for an example  $x \in \mathcal{X}$  with three views. At each time step  $t \in \{0, 1, \dots, T\}$ , representations  $x_t^{(k)}$  for each  $k \in \{1, 2, 3\}$  views are fed to a cell. The cell outputs a joint representation  $h_t$  for all the views.

modifications pertaining to early fusion. For example, they may operate on inputs consisting of the aggregation of representations coming from different sources. Indeed, owing to its simplicity, and, sometimes, acceptable performance, this way is taken in many applications. Moreover, since multi-view fusion is an open research question, the use of this early fusion is imperative to enable comparison between models that do not differ in the fusion method. For instance, a novel model designed for the problem of emotion identification is more directly comparable to other models proposed for the same problem if they use the same fusion method.

Arguably all the points in favour of view fusion discussed in the Introduction and the previous sections hold true for sequential input as well. Thus, models utilizing the RNN framework and borrowing ideas from common models such as the LSTM or GRU have been proposed to account for multi-view inputs. From a blackbox point of view, such models operate on data points  $x \in \mathcal{X}$  consisting of elements  $x_t^{(k)}$ , for each timestep  $t \in \{0, 1, 2, \dots, T\}$  and view  $k \in \{1, 2, \dots, K\}$ . A general overview of such a model is given in Figure 2.II.

### 2.3.3 Non-correlational multi-view sequential approaches

A large set of multi-view sequence models does not have a correlational orientation; they incorporate information from several sources in different ways. The relevant techniques fall broadly in two main categories: *external*, that is mechanisms built around sequence models, and *internal*, that is approaches that propose fundamental changes in the sequence model cells to add multi-view functionality.

In the *external* domain, the multi-view functionality can be entrusted to layers previous to the sequence model. For example, the work of Cui et al. (2018a) proposes feeding an LSTM with fused representations created by an autoencoder. In a similar fashion, multimodal em-

beddings constructed in a way that the main modality (for example, text) does not get downgraded by other noisy modalities (image or acoustic), can be fed to a sequence model. This approach is taken in the work of [Chen et al. \(2017\)](#), where a Gated Multimodal Embedding is constructed at the word level and subsequently fed to an LSTM.

External approaches also include attention mechanisms operating on the output of sequence models. The work of [Zadeh et al. \(2018a\)](#) employs an attention mechanism that fuses representations from three different sequence models. Similarly, [Zadeh et al. \(2018b\)](#) employ an attention mechanism (Multimodal State Memory) informed by representations created from the output of three different LSTMs (one for each of the modalities present in the problem) which are fused by the novel fusion method of the Dynamic Fusion Graph.

Approaches belonging to the *internal* domain mostly focus on devising modifications to either the LSTM or the GRU. Specifically, considering the defining equations of the LSTM (Equations 2.24 through 2.25), the simplest multi-view variant would be to employ a separate LSTM for each of the available views. Such a model would be described by the following equations:

$$\begin{aligned}
 f_t^{(k)} &= \sigma_g(W_f^{(k)} x_t^{(k)} + U_f^{(k)} h_{t-1}^{(k)} + b_f^{(k)}) \\
 i_t^{(k)} &= \sigma_g(W_i^{(k)} x_t^{(k)} + U_i^{(k)} h_{t-1}^{(k)} + b_i^{(k)}) \\
 \tilde{c}_t^{(k)} &= \sigma_c(W_c^{(k)} x_t^{(k)} + U_c^{(k)} h_{t-1}^{(k)} + b_c^{(k)}) \\
 c_t^{(k)} &= f_t^{(k)} \circ c_{t-1}^{(k)} + i_t^{(k)} \circ \tilde{c}_t^{(k)} \\
 o_t^{(k)} &= \sigma_o(W_o^{(k)} x_t^{(k)} + U_o^{(k)} h_{t-1}^{(k)} + b_o^{(k)}) \\
 h_t^{(k)} &= o_t^{(k)} \circ \sigma_h(c_t^{(k)}),
 \end{aligned} \tag{2.27}$$

with  $k$  iterating through the views. Note how in these equations everything is view-specific (weights, biases, inputs). Slightly relaxing this constraint gives rise to view-centric representations, that are nonetheless informed by the rest of the views. This is the case with the Multimodal LSTM proposed by [Ren et al. \(2016\)](#). In order to create a multi-view model, they force weight sharing across views for  $U_f^{(k)}$ ,  $U_i^{(k)}$ ,  $U_c^{(k)}$  and  $U_o^{(k)}$ , thus replacing those weights with the common for all views  $U_f$ ,  $U_i$ ,  $U_c$  and  $U_o$ .

In their work, [Rajagopalan et al. \(2016\)](#) introduce multi-view LSTM (MV-LSTM), a multi-view variant of LSTM and at the same time present a framework to build multi-view sequential models with a degree of flexibility. Essentially, the MV-LSTM cell incorporates a separate LSTM for each of the views, in the gates of which, some information from the

other views is passed on. The degree of reliance to the other views is controlled by two hyperparameters  $\alpha$  and  $\beta$ , taking values between 0 and 1, which account for the amount of information that should be view-specific and input-oriented respectively. For example, a value of  $(\alpha, \beta) = (1/3, 1/3)$  indicates that the memory should contain view-specific, input-oriented and coupled cells, while a value of  $(\alpha, \beta) = (1, 0)$  results in a fully view-specific model: each view interacts with previous representations of the same view only. Finally, a value of  $(\alpha, \beta) = (0, 1)$  results in a coupled model: each view interacts with previous representations of other views only. The values of the hyperparameters are incorporated into the LSTM equations through matrices  $A$  and  $B$ :

$$\begin{aligned} A_{ij} &= \begin{cases} 1, & i = j, i \leq \alpha \\ 0, & \text{otherwise} \end{cases} \\ B_{ij} &= \begin{cases} 1, & i = j, i \geq (1 - \beta) \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \quad (2.28)$$

Figure 2.12 presents the architecture of the MV-LSTM cell. It is worth noting that although there is an interaction between views in the memory cell, the core functionality revolves around separate representations for each of the views. The MV-LSTM for a set  $\mathcal{V}$  of views, is described by the following equations:

$$\begin{aligned} f_t^{(k)} &= \sigma_g(W_{fx}^{(k)} x_t^{(k)} + U_{fx}^{(k)} A h_{t-1}^{(k)} + \sum_{\substack{m=1 \\ m \neq k}}^{|\mathcal{V}|} W_{fh}^{(m)} B h_{t_1}^{(m)}) \\ i_t^{(k)} &= \sigma_g(W_{ix}^{(k)} x_t^{(k)} + U_{ix}^{(k)} A h_{t-1}^{(k)} + \sum_{\substack{m=1 \\ m \neq k}}^{|\mathcal{V}|} W_{ih}^{(m)} B h_{t_1}^{(m)}) \\ o_t^{(k)} &= \sigma_g(W_{ox}^{(k)} x_t^{(k)} + U_{ox}^{(k)} A h_{t-1}^{(k)} + \sum_{\substack{m=1 \\ m \neq k}}^{|\mathcal{V}|} W_{oh}^{(m)} B h_{t_1}^{(m)}) \\ \tilde{c}_t^{(k)} &= \sigma_h(W_{cx}^{(k)} x_t^{(k)} + U_{cx}^{(k)} A h_{t-1}^{(k)} + \sum_{\substack{m=1 \\ m \neq k}}^{|\mathcal{V}|} W_{ch}^{(m)} B h_{t_1}^{(m)}) \\ c_t^{(k)} &= f_t^{(k)} \circ c_{t-1}^{(k)} + i_t^{(k)} \circ \tilde{c}_t^{(k)} \\ h_t^{(k)} &= o_t^{(k)} \circ c_t^{(k)}. \end{aligned} \quad (2.29)$$

Ultimately, there are methods that do not strictly fall in neither of the two categories. The

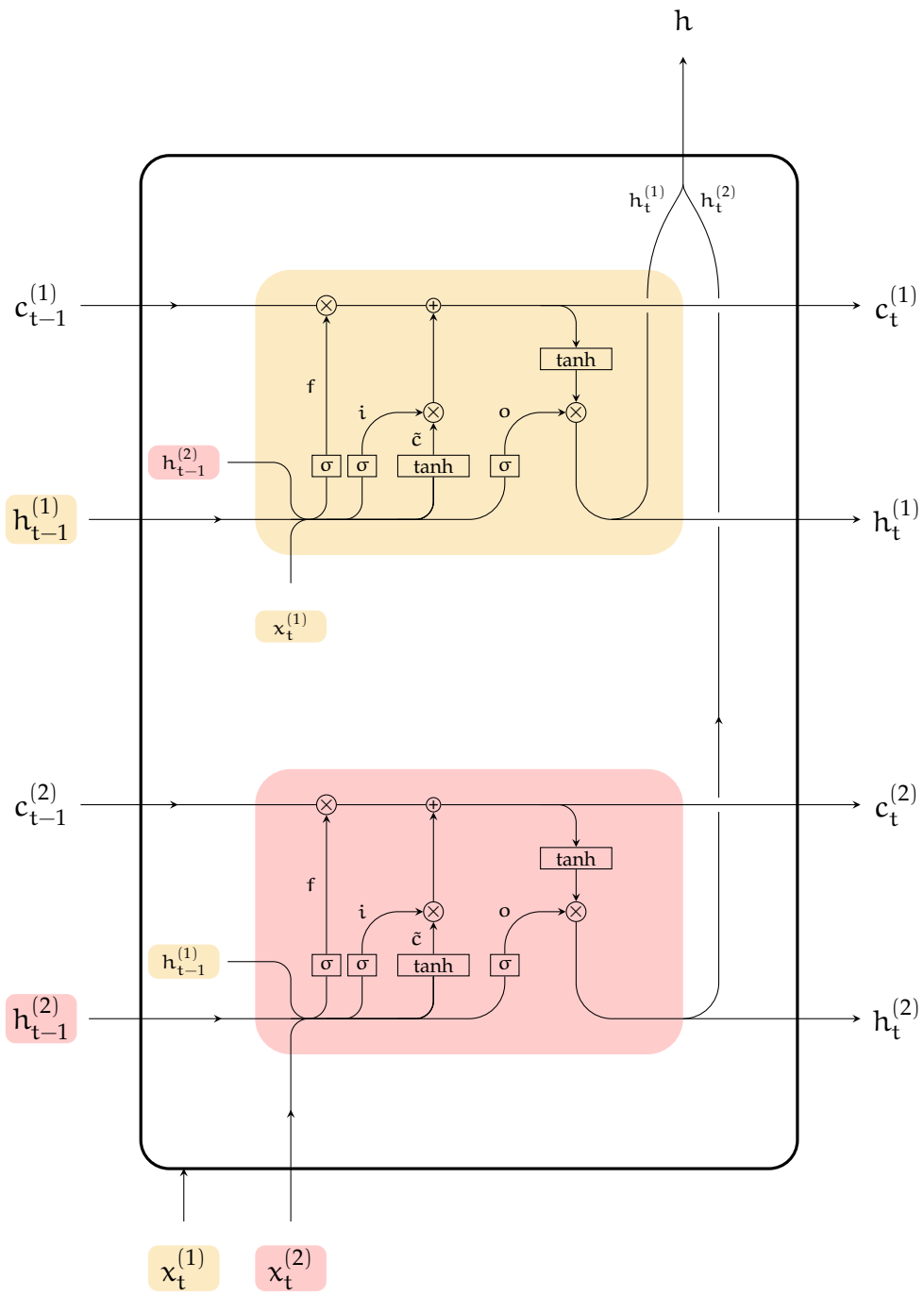


Figure 2.12: Architecture of a Multi-view LSTM (MV-LSTM) cell.

work of Liang et al. (2018), although not presented as such, is more relevant to the internal domain than the external. It uses a mechanism that is interjected between consecutive runs of a sequence model and alters the representations carried to the next step. This mechanism, called Multistage Fusion Process uses three different modules (HIGHLIGHT, FUSE and SUMMARIZE) that repeatedly identify and locally fuse subsets of multimodal signals. The fusion process can be thought of as being integrated in one, multi-view cell, thus corroborating the internal nature of this model.

### 2.3.4 Correlational multi-view sequential approaches

One predominantly correlational sequence model is the correlational Gated Recurrent Unit (corrGRU; Yang et al. 2017); a multi-view extension of the GRU. In theory, the simplest multi-view GRU would be one that uses a separate GRU cell for each view. Similarly to the simple multi-view LSTM (Equations 2.27) and with respect to the structural equations of a GRU (Equations 2.26), such a model would be described by the following equations:

$$\begin{aligned}
 z_t^{(k)} &= \sigma_g(W_z^{(k)} x_t^{(k)} + U_z^{(k)} h_{t-1}^{(k)} + b_z^{(k)}) \\
 r_t^{(k)} &= \sigma_g(W_r^{(k)} x_t^{(k)} + U_r^{(k)} h_{t-1}^{(k)} + b_r^{(k)}) \\
 \tilde{h}_t^{(k)} &= \sigma_h(W_h^{(k)} x_t^{(k)} + U_h^{(k)} (r_t^{(k)} \circ h_{t-1}^{(k)}) + b_h^{(k)}) \\
 h_t^{(k)} &= (1 - z_t^{(k)}) \circ h_{t-1}^{(k)} + z_t^{(k)} \circ \tilde{h}_t^{(k)},
 \end{aligned} \tag{2.30}$$

with  $k$  iterating through the views.

The corrGRU takes a slightly different approach: it creates and uses a single hidden state for all modalities. This is made possible by fusing representations within the cell. To achieve that, it first employs a simple GRU for each of the views, wherein the common hidden representation from the previous step is used:

$$\begin{aligned}
 z_t^{(k)} &= \sigma_g(W_z^{(k)} X_t^{(k)} + U_z h_{t-1} + b_z^{(k)}) \\
 r_t^{(k)} &= \sigma_g(W_r^{(k)} X_t^{(k)} + U_r h_{t-1} + b_r^{(k)}) \\
 \tilde{h}_t^{(k)} &= \sigma_h(W_h^{(k)} X_t^{(k)} + U_h (r_t^{(k)} \circ h_{t-1}) + b_h^{(k)}) \\
 h_t^{(k)} &= (1 - z_t^{(k)}) \circ h_{t-1} + z_t^{(k)} \circ \tilde{h}_t^{(k)},
 \end{aligned} \tag{2.31}$$

and, subsequently, implements multi-view gates by using a weighted sum of the view-specific

representations:

$$\begin{aligned}
z_t &= \sigma_g \left( \sum_{k=1}^{|\mathcal{V}|} w_t^{(k)} (W_z^{(k)} X_t^{(k)} + b_z^{(k)}) + U_z h_{t-1} \right) \\
r_t &= \sigma_g \left( \sum_{k=1}^{|\mathcal{V}|} w_t^{(k)} (W_r^{(k)} X_t^{(k)} + b_r^{(k)}) + U_r h_{t-1} \right) \\
\tilde{h}_t &= \sigma_h \left( \sum_{k=1}^{|\mathcal{V}|} w_t^{(k)} (W_h^{(k)} X_t^{(k)} + b_h^{(k)}) + U_h (r_t^{(k)} \circ h_{t-1}) \right) \\
h_t &= (1 - z_t) \circ h_{t-1} + z_t \circ \tilde{h}_t,
\end{aligned} \tag{2.32}$$

with  $k$  iterating over the set  $\mathcal{V}$  of views.<sup>5</sup> The weights  $w_t^{(k)}$  are generated by a dynamic weighting module (DW), which outputs weights for each time step by considering its coherence with the fused representation of the previous time step. Consequently, the model is able to focus more on modalities that carry informative signal that is consistent with previous timesteps, as opposed to, for example, modalities corrupted by noise. As such, dynamic weighting can be regarded as a loose form of attention. More specifically, coherence scores  $\alpha_t$  for each view  $k$  and time step  $t$  are calculated as

$$\alpha_t^{(k)} = x_t^{(k)} A_k h_{t-1}^\top, \tag{2.33}$$

where  $A_k$  are parameters learnt and the weights are then obtained by applying Laplace smoothing:

$$w_t^{(k)} = \frac{1 + \exp(\alpha_t^k)}{2 + \sum_m \exp(\alpha_t^{(m)})}, \tag{2.34}$$

with  $k$  iterating through the views. Dynamic weighting is reported to have significant impact in multimodal fusion in the work of Yang et al. (2017); a claim consistent with our experiments (see Chapter 5).

A comprehensive diagram of the architecture of the corrGRU can be seen in Figure 2.13. What makes the corrGRU actually correlational, is the fact that it encapsulates a correlation term in its objective, thus forcing the maximisation of the correlation between the representations for each view. Formally, assuming just two views, in each step the Pearson correlation

<sup>5</sup>In the paper of Yang et al. (2017), only two views are considered ( $|\mathcal{V}| = 2$ ), but here we present a more general form of the equations.



between them is calculated as

$$c_t(H_t^{(1)}, H_t^{(2)}) = \frac{\sum_{i=1}^L (h_{it}^{(1)} - \bar{H}_t^{(1)})(h_{it}^{(2)} - \bar{H}_t^{(2)})}{\sqrt{\sum_{i=1}^L (h_{it}^{(1)} - \bar{H}_t^{(1)}) \sum_{i=0}^L (h_{it}^{(2)} - \bar{H}_t^{(2)})}}, \quad (2.35)$$

where  $i$  spans over the  $L$  elements of each mini-batch,  $h_{it}^{(k)}$  is the hidden state calculated by the view-specific GRU for the example  $i$ , view  $k$ , at timestep  $t$ . Moreover,  $\bar{H}_t^{(k)} = \frac{1}{L} \sum_{i=1}^L h_{it}^{(k)}$ .

The correlation term is then defined as

$$\mathcal{L}_{\text{corr}} = \frac{1}{|\mathbb{T}|} \sum_{t \in \mathbb{T}} c_t, \quad (2.36)$$

that is the average of the loss calculated for every time step  $t$ . Generalising to more than two views, one can calculate the total correlation as the sum of the correlation between all pairs of elements of  $\mathcal{V}$ . The network is consequently trained by adding the correlation loss term to any other loss terms used. Formally,

$$\mathcal{L} = \mathcal{L}_{\text{other}} - \lambda \mathcal{L}_{\text{corr}}, \quad (2.37)$$

where  $\lambda$  is a hyperparameter weighing the contribution of  $\mathcal{L}_{\text{corr}}$  to the total loss. The negative sum is used because we would like the correlation to be maximised, while normally losses express quantities to be minimised. A general overview of the corrGRU model is shown in Figure 2.13.

Yang et al. (2017) report experiments with corrGRU on action classification with video sensor data with the ISI dataset (Kumar et al., 2015) and audio-visual speech classification with AVLetters (Matthews et al., 2002) and CUAVE (Patterson et al., 2002) datasets. We experiment with using corrGRU for multimodal applications that include a strong language component and more than two modalities (Chapters 5 and 6).

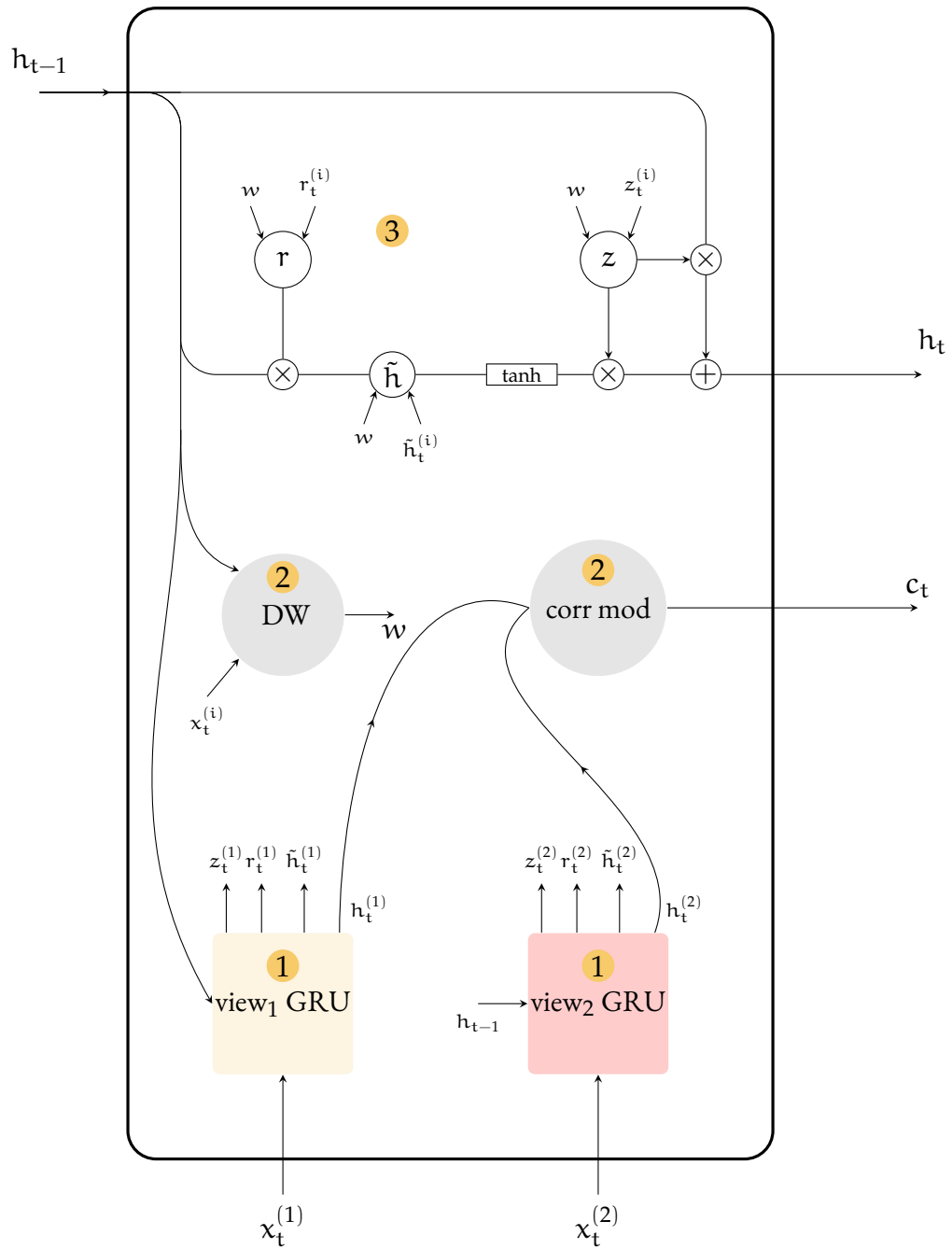


Figure 2.13: Architecture of the correlational GRU (corrGRU) cell. The numbers in circles correspond to the order in which the calculations are performed: first, each view is fed to a simple GRU cell, then the output of those cells is used to calculate the weight scores (Dynamic Weighting module; DW) and the correlation between the views (correlation module; corr mod). Lastly, a joint representation is calculated for all views, which will be fed to the next timestep. In the figure, for clarity, several links are omitted, but the labels of the arrows define where the links should be. For example, after the first step, the values  $h_t^{(1)}, h_t^{(2)}, \dots$  are fed to the correlation module as  $h_t^{(i)}$ .



## Chapter 3

# Multi-view Document Summarisation

*Information overload is a state in which a decision maker faces a set of information [...] comprising the accumulation of individual informational cues of differing size and complexity that inhibit the decision maker's ability to optimally determine the best possible decision.*

*(Roetzzel, 2018)*

Information overload is a state affecting not only decision makers in the strict political sense, but also each user of media, newspapers, web portals or social media. Although there is no universally acceptable definition of information overload, the technology available to any “decision maker” or user plays an important role when establishing its existence. This has led in conflicting opinions on the issue: one that views information overload as a problem (Dean and Webb, 2011) and one viewing it as an opportunity to inspire technological advances (Blair, 2012). Search engines are an example of technological product that has boosted the ability of users to cope with massive amounts of information and caused a shift in the perception of information overload.

The abundance of textual information online has brought about a growing demand for NLP expertise. Automatic summarisation systems are just one example of technological product that relies heavily on language technology and can help users navigate through the information overload they face in a daily basis. Specifically, using an automatic summariser, a user can quickly and effortlessly digest documents, spotting major events, entities' involvement in events or emerging trends in the media, without having to spend a substantial amount of time reading. The NLP community has been actively trying to fill this user need, by researching and developing automatic summarisation methods and techniques.

Naturally, the need for automatic summarisation is not constrained to personal usage. News

agencies such as the British Broadcasting Corporation (BBC) employ monitoring agencies (BBC Monitoring division) that track television, internet and social media sources on a daily basis, with the goal of detecting trends or changes in media outlets, or spot and flag breaking news events quickly and accurately (Liepins et al., 2017). Analysts in such agencies are faced with the problem of ingesting massive amounts of information in short time, from diverse and multilingual sources. The same is also true for internal consumption of news items in large news broadcasters, such as the Deutsche Welle (DW). From this point of view, the need for summarisation systems that can accommodate more than one languages and domains of text is imperative.

Commonly, news consumers are presented with multifarious and multimodal information when accessing a news portal. Although paper-based media still are in existence and publication, it is apparent that there has been a shift towards websites and social media, especially for younger audiences.<sup>1</sup> Articles employ a mixture of text, images, videos, diagrams, links to other articles, reader comments or social media posts that are relevant to the article. Consequently, a multi-view approach, where all or some of this side information is exploited to enrich NLP models, is appealing. Indeed, using extra information has been shown to benefit summarisation systems (Narayan et al., 2018a), even outside a multi-view environment.

In this chapter, we describe a multi-view take on document summarisation, that of regarding an article and its summary as two views of the same input. The underlying assumption is that there is a latent meaning in the article (a meaning that conveys the semantics of the text), whose observed variables are the text and the summary. This assumption leads to the use of multi-view techniques, such as Canonical Correlation Analysis to attack the problems of extractive and abstractive summarisation. Most of the work described in this chapter does not use any other modalities or other extra information or metadata surrounding articles, apart from the text of the documents and the summaries. Thus, the described work constitutes a solid example of work using multi-view techniques, without having a multimodal motive.

### 3.1 Document Summarisation

The process of distilling the knowledge of a text passage in a shorter, paraphrased text is a fundamental element of reading and writing skills of humans. Its importance relies on the utility of the product, since a summary is easier and quicker to read than a full text. Moreover, it also relies on the effects of the process; it has been suggested that summarising

---

<sup>1</sup>For trends in the U.S.A., consult the report of Mitchell et al. (2016).

is an effective way to learn from text (Kissner, 2006). Interestingly, summarisation is a hard task, even for humans (Hill, 1991).

Within the NLP community, automatic document summarisation is a quite popular and extensively studied problem. Partly due to the extreme difficulty of the problem and partly due to the insurmountable obstacle of not being able to have a concrete, exactly right solution to look up on while training summarisation models, there have been several simplification attempts to the problem of automatic summarisation. The most popular simplification is the introduction of *extractive summarisation*; a proxy problem suggesting that a summary can be created by selecting a handful of sentences from the original text passage, instead of creating a novel summary. Extractive summarisation stands on the opposite side of what is called *abstractive summarisation*, which is the more natural or intuitive way of summarising, involving the generation of a new synopsis that paraphrases the original input's words, from scratch.

Automatic summarisation is broad, and can include tasks that agreeably should not be treated as equal or, even, similar. Single document summarisation refers to generating a summary from one text passage, regardless of its size: a twenty sentence article and a twenty page report are, in essence, one document each. Moreover, depending on the application, generated summaries can be general-purpose (including the main ideas or events of the text) or query-based (focused on a specific query that comes from a user). All the work that is described in the present chapter concerns the generation of single-document, general-purpose summaries.

### 3.1.1 Extractive Summarisation

Extractive summarisation refers to casting summarisation as a selection problem. Specifically, for a document that consists of  $n \in \mathbb{N}^+$  sentences, the extractive summarisation process will select  $k < n, k \in \mathbb{N}^+$  sentences which are the most representative of the text. The output of the extractive summarisation system is the concatenation of the extracted sentences. An example of an extractive summary can be seen in Figure 3.1.

Extractive summarisation has been thoroughly investigated, and as such, to date, extractive systems are able to produce output of good or acceptable quality. A reason for the guaranteed quality of an extractive summariser stems from the fact that it selects sentences from the document, thus ensuring that the summary will consist of sentences that are as grammatical and well-formed as the input. At the same time though, the output text may be incoherent,

since the extractive framework does not necessarily account for joining the sentences in an elegant way, or ensuring that two or more sentences do not contain the same information.

Extractive systems for various tasks (single- document or multi- document, general or query-based summarisation) may rely on manually engineered features or shallow text features, such as the length and position of sentences (Radev et al., 2004), the presence of specific important cue words, title words or word frequencies (Nenkova et al., 2006). Some works also attempt deep understanding of documents before summarising, incorporating discourse features (Osborne, 2002) or even event information (Filatova and Hatzivassiloglou, 2004). The selection framework varies from greedy techniques (for example, the work of Wan et al. 2007) and graph techniques (Erkan and Radev, 2004) to constraint optimisation (such as the work of Alguliev et al. 2011).

Recent advances in neural models have propelled the creation of data-driven systems that do not require great amount of feature engineering (examples include the works of Yin and Pei 2015; Nallapati et al. 2016a; Narayan et al. 2018a). Although the recent resurgence in interest for summarisation has risen the bar for state-of-the-art performance of extractive summarisers, oracle experiments show that there is still room for improvement (Hirao et al., 2017).

### 3.1.2 Abstractive Summarisation

Although there is no consensus on an absolutely correct way to summarise texts,<sup>2</sup> it is apparent that humans summarise in a profoundly different way from extractive summarisers. A human subject asked to summarise a text passage, will, most probably try to paraphrase the original story in their own words, while trying to work on an abstract level. The generated summary will include some, but not all, the details conveyed in the text passage, while some may be mentioned briefly (for example, explicit numbers such as “2.5 million people from a country with a 5 million population” may end up being mentioned as “50% of the country’s population”, “half of the country’s population” and so on). This way of creating summaries, by paraphrasing the original text is called *abstractive summarisation*, emphasising on the fact that the generated summary will contain the abstract idea of the document. An abstractive summary seldom includes verbatim sentences from the document and since it is created with the goal of being a self-contained text, it is in principle coherent and does not contain redundant information.

---

<sup>2</sup>There are some sets of guidelines that can be taught to students, though (Brown et al., 1981).

Figure 3.1 shows an example of an article from the news portal of Cable News Network (CNN), along with three different summaries. The summaries listed are:

- *LEAD-3*: a summary constructed by concatenating the three first sentences of the article.
- *extractive*: the output of an extractive summarisation system (SideNet; Narayan et al. 2018a), consisting of three sentences of the article.
- *abstractive*: the summary taken from the CNN website, which has been generated by human reporters. The CNN website,<sup>3</sup> for a period of time, used to include *highlights* in the form of bullet points that summarise the main contents of articles. Highlights were listed on the top of the article’s webpage to aid users who would like to be informed without going through the article in detail. While this specific highlight format does not account for a coherent text, the term “abstractive” about the summary is justified by the fact that highlight sentences are original (not copied from the document).

While the task of automatic abstractive summarisation has been standardised in the DUC-2003 and DUC-2004 challenges,<sup>4</sup> it has seen a resurgence in interest lately, partially fueled by advances on neural sequence models (Rush et al., 2015; Nallapati et al., 2016b). Proposed approaches on abstractive summarisation have been quite diverse, with techniques relying on machine translation frameworks, either phrase-based (such as the work of Banko et al. 2000) or sequence-to-sequence (seq2seq) models (for example, the work of Hu et al. 2015), compression-based approaches using weighted tree-transformation rules (Cohn and Lapata, 2008) and quasi-synchronous grammar approaches (Woodsend et al., 2010). Neural abstractive summarisers are usually developed around the seq2seq framework, but do not fail to include several sophisticated modules, such as attention (Rush et al., 2015) or copying mechanisms (Nallapati et al., 2016b).

#### **Mosquito-borne virus chikungunya worries CDC**

A debilitating, mosquito-borne virus called chikungunya has made its way to North Carolina, health officials say. It’s the state’s first reported case of the virus.

The patient was likely infected in the Caribbean, according to the Forsyth County Department of Public Health. Chikungunya is primarily found in Africa, East Asia and the Caribbean islands, but the Centers for Disease Control and Prevention has been watching the virus, for fear that it could take hold in the United States – much

<sup>3</sup><https://edition.cnn.com/>

<sup>4</sup><http://duc.nist.gov/>



like West Nile did more than a decade ago.

The virus, which can cause joint pain and arthritis-like symptoms, has been on the U.S. public health radar for some time. About 25 to 28 infected travelers bring it to the United States each year, said Roger Nasci, chief of the CDC's Arboviral Disease Branch in the Division of Vector-Borne Diseases.

"We haven't had any locally transmitted cases in the U.S. thus far," Nasci said.

But a major outbreak in the Caribbean this year – with more than 100,000 cases reported – has health officials concerned. Experts say American tourists are bringing chikungunya back home, and it's just a matter of time before it starts to spread within the United States.

After all, the Caribbean is a popular one with American tourists, and summer is fast approaching.

"So far this year we've recorded eight travel-associated cases, and seven of them have come from countries in the Caribbean where we know the virus is being transmitted," Nasci said.

Other states have also reported cases of chikungunya. The Tennessee Department of Health said the state has had multiple cases of the virus in people who have traveled to the Caribbean.

The virus is not deadly, but it can be painful, with symptoms lasting for weeks. Those with weak immune systems, such as the elderly, are more likely to suffer from the virus' side effects than those who are healthier.

The good news, said Dr. William Shaffner, an infectious disease expert with Vanderbilt University in Nashville, is that the United States is more sophisticated when it comes to controlling mosquitoes than many other nations.

"We live in a largely air-conditioned environment, and we have a lot of screening (window screens, porch screens)," Shaffner said. "So we can separate the humans from the mosquito population, but we cannot be completely be isolated."

Chikungunya was originally identified in East Africa in the 1950s. The ecological makeup of the United States supports the spread of an illness such as this, especially in the tropical areas of Florida and other Southern states, according to the CDC.

The other concern is the type of mosquito that carries the illness. Unlike most mosquitoes that breed and prosper outside from dusk to dawn, the chikungunya virus is most often spread to people by *Aedes aegypti* and *Aedes albopictus* mosquitoes.

These are the same mosquitoes that transmit the virus that causes dengue fever. They bite mostly during the daytime. The disease is transmitted from mosquito to human, human to mosquito and so forth. A female mosquito of this type lives three to four weeks and can bite someone every three to four days.

Shaffner and other health experts recommend people remember the mosquito-control basics:

- Use bug spray if you are going out, especially in tropical, or wooded areas near water.
- Get rid of standing water, empty plastic pools, flower pots and pet dishes, so mosquitoes don't breed.
- Dress appropriately, with long sleeves and pants.

---

### **LEAD summary**

- A debilitating , mosquito-borne virus called chikungunya has made its way to

North Carolina , health officials say.

- It 's the state 's first reported case of the virus.
- The patient was likely infected in the Caribbean, according to the Forsyth County Department of Public Health.

---

**Extractive summary**

- A debilitating , mosquito-borne virus called chikungunya has made its way to North Carolina, health officials say.
- Chikungunya is primarily found in Africa, East Asia and the Caribbean islands, but the Centers for Disease Control and Prevention has been watching the virus, for fear that it could take hold in the United States – much like West Nile did more than a decade ago.
- About 25 to 28 infected travelers bring it to the United States each year, said Roger Nasci, chief of the CDC's Arboviral Disease Branch in the Division of Vector-Borne Diseases.

---

**Human-generated (abstractive) summary**

- North Carolina reports first case of mosquito-borne virus called Chikungunya.
- Chikungunya is primarily found in Africa, East Asia and the Caribbean islands.
- Virus is not deadly, but it can be painful, with symptoms lasting for weeks.

Figure 3.1: An article from the CNN website (<https://edition.cnn.com/2014/06/12/health/virus-chikungunya/>, accessed March 10, 2019.), with three summaries: the LEAD-3 summary, the output of an extractive summarisation system (SideNet) and the original summary it is paired with in the CNN website. All summaries are presented as lists because none of them was written as a coherent text.

### 3.1.3 Evaluation

Commonly, the output of summarisation systems is compared with that from a simple baseline, called the LEAD baseline. The LEAD- $k$  baseline generates summaries by concatenating the first  $k$  sentences of the document. The intuition behind this baseline is that the very first sentences of a document contain important information, possibly the most important of the whole document. Although this is empirical observation, it can be traced back to writing strategies (as newspaper editors say “don't bury the lede”, meaning “put the important things first”; [Pinker 2015](#)) or to standard journalistic practice (the first sentences are read by more people than the whole article). It turns out that the LEAD baseline is strong, suggesting that a large amount of important information indeed lies at the beginning of the document.

While deciding on the value of  $k$  may involve some flexibility and is also domain- or task-dependent, research work in news articles usually uses a value of 3, possibly due to the size of

the generated summary (a two-sentence summary may be quite short, while a four-sentence long, considering the average length of an article). Moreover, popular news portals such as the CNN<sup>5</sup> and DailyMail<sup>6</sup> seem to use (or have used) three-sentence summaries (or highlights).

### 3.1.3.1 Automatic Evaluation

Efficient and easily available evaluation is of utmost importance for NLP and machine learning tasks. Their existence makes development of new methods easier, as they provide a robust way to compare between variants of models, pointing to promising directions or guiding hyperparameter tuning. However, automatic evaluation of NLP tasks, especially those involving a text generation or selection part, is not straightforward.

Recall-Oriented Understudy for Gisting Evaluation (ROUGE; Lin 2004) proposes a set of metrics that can be easily calculated and are tailored to the problem of summarisation. ROUGE scores rely on the amount of common language units (n-grams) between a summary that has been generated by a system (“system summary”) and a correct (“reference”) summary. The ROUGE framework also provides a way to compare a generated summary to more than one reference summaries, to countermeasure the fact that for a given text passage, there can be more than one, equally good or adequate summaries.

Specifically, ROUGE includes the following metrics:

- **ROUGE-N** refers to a set of co-occurrence based metrics. The value of N determines the type of language unit considered (ROUGE-1 operates on unigrams, ROUGE-2 on bigrams and so on). Formally, the ROUGE-N score between a system summary S and a set of reference summaries  $\mathcal{R}$  is defined as:

$$\text{ROUGE-N} = \frac{\sum_{s' \in \mathcal{R}} \sum_{u_N \in s'} M(u_N, S)}{\sum_{s' \in \mathcal{R}} \sum_{u_N \in s'} C(u_N)}, \quad (3.1)$$

where C denotes the count function and M a function returning 1 if the unit  $u_N$  occurs in summary S and 0 otherwise. Varying the value of N changes the informativity of the metric accordingly. For example, since ROUGE-1 simply counts co-occurrence of unigrams, it may not be too indicative of the quality of the generated summary (it is equivalent to bag-of-words intersection between the system and the reference summary). On the other hand, ROUGE-4, which counts co-occurrence of four-grams,

---

<sup>5</sup><https://edition.cnn.com/>

<sup>6</sup><https://www.dailymail.co.uk/>

may be too specific and, consequently, difficult to achieve high values. Commonly, summarisation works mention at least ROUGE-1 and ROUGE-2 scores.

- **ROUGE-L** accounts for the longest common subsequence between the generated summary and a reference summary. Contrary to ROUGE-N, ROUGE-L considers co-occurrence of language units, but only in-sequence, albeit not necessarily consecutive. Given two sequences of tokens  $X$  and  $Y$  with lengths  $|X|$  and  $|Y|$  respectively, we can calculate the following:

$$r_L = \frac{\text{LCS}(X, X')}{|X|} \quad (3.2)$$

$$p_L = \frac{\text{LCS}(X, X')}{|X'|} \quad (3.3)$$

$$f_L = \frac{(1 + \beta^2)r_L p_L}{r_L + \beta^2 p_L}, \quad (3.4)$$

where  $r_L$  is recall,  $p_L$  precision,  $f_L$  the corresponding  $f$ - $\beta$  measure and LCS the longest common subsequence. The same scores calculated for reference summary  $S$  consisting of  $|S|$  tokens and  $u$  sentences and system summary  $S'$  with  $|S'|$  tokens, are calculated as follows:

$$r_L = \frac{\sum_{i=1}^u \text{LCS}_U(S_i, S')}{|S|} \quad (3.5)$$

$$p_L = \frac{\sum_{i=1}^u \text{LCS}_U(S_i, S')}{|S'|}, \quad (3.6)$$

where  $\text{LCS}_U$  denotes the union longest common subsequence. To illustrate the calculation of  $\text{LCS}_U$ , consider the following example: given a reference sentence  $r_i = [w_1 w_2 w_3 w_4 w_5]$  and a system summary with two sentences  $c_1 = [w_1 w_2 w_6 w_7 w_8]$  and  $c_2 = [w_1 w_3 w_8 w_9 w_5]$ , with  $w_j$  being tokens, the longest common subsequence between  $r_i$  and  $c_1$  is  $[w_1 w_2]$  and between  $r_i$  and  $c_2$ ,  $[w_1 w_3 w_5]$ . The union longest common subsequence considers the union of the two:  $[w_1 w_2 w_3 w_5]$ , thus  $\text{LCS}_U(r_i, C) = 4$ .

- the ROUGE suite also includes other metrics, such as **ROUGE-S** (co-occurrence of skip-bigrams, i.e. pairs of words in their sentence order, allowing for arbitrary gaps),

**ROUGE-SU** (ROUGE-S variant that takes also unigrams into consideration) and **ROUGE-W** (ROUGE-L variant where the number of common subsequences that consist of consecutive elements is considered).

The software package released for the ROUGE suite<sup>7</sup> is able to calculate and report all the above scores. Moreover, it provides an option to generate scores for specific summary lengths. Partially consulting summaries leads to normalised evaluation that does not unfairly favor longer summaries over shorter ones. This is especially useful for the evaluation of extractive summarisers, that have no inherent way to restrict the length of the generated summary. Commonly, evaluation is performed for full length (no length limit), 275 bytes (the first 275 bytes of the summary are taken into consideration) and 75 bytes (roughly one sentence).

Importantly, in the literature for extractive summarisation, recall metrics are reported, since, in the extractive setup, sentences are copied verbatim from the output and there is no way for the summariser to control for precision. In abstractive summarisation, though, all of recall, precision and F scores are reported, since the summariser has more degrees of freedom and absolute control over the output. Additionally, natural language generation metrics, such as METEOR (Denkowski and Lavie, 2014) have been reported for abstractive summarisation (See et al., 2017), although the use of ROUGE is prevalent in the literature.

### 3.1.3.2 Human Evaluation

Despite their convenience, automatic metrics have limited capacity. Traditionally, metrics themselves are evaluated by calculating the correlation between their score values and human judgments. The problem of creating the “best” metric, or selecting one of the available metrics that behaves the best, is an open problem. Although ROUGE has been adopted in large by the NLP community, there have been several re-assessments of its soundness and suitability for summarisation evaluation (Lin and Och, 2004; Dorr et al., 2005; Graham, 2015).

Evaluation of the output of summarisers by human subjects can ensure that the variability and fallibility of automatic metrics can be overcome. Despite the high cost it may incur (both in time and money), human evaluation has been included in several summarisation works (such as the work of Cheng and Lapata 2016 and that of Narayan et al. 2018b), as a more fine-grained evaluation than automatic metrics or as a sanity check to state-of-the-art results.

---

<sup>7</sup>Available from <https://github.com/andersjo/pyrouge/tree/master/tools/ROUGE-1.5.5>.

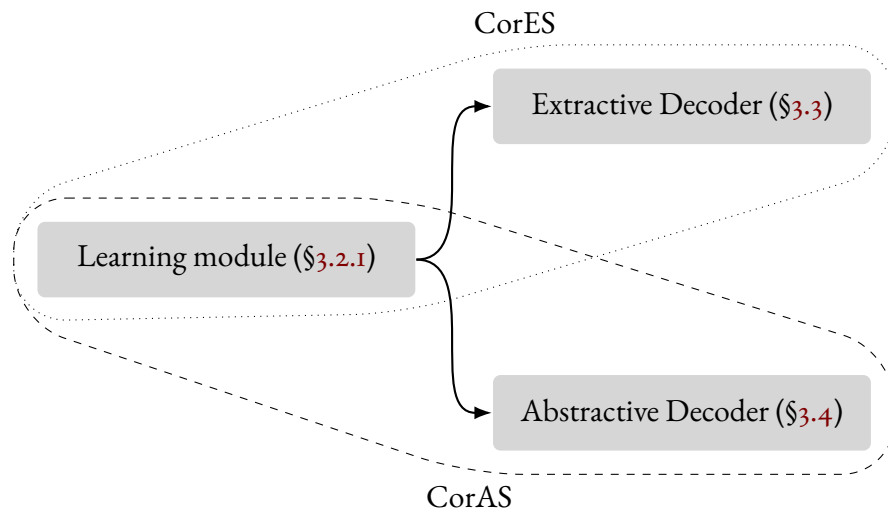


Figure 3.2: Overview of the proposed summarisation systems. The learning module is used by both the extractive summariser (CorES) and the abstractive summariser (CorAS).

Potential shortcomings of human evaluation include poor evaluation standards, such as small sample (justified by the cost constraints of conducting human evaluation) or the absence of community-wide standards (if human evaluations are not conducted in a similar way, or using different criteria to rank summaries, their output may not be comparable to other works).

## 3.2 Multi-view Representation Learning for Document Summarisation

The present chapter describes a multi-view approach for single-document summarisation. The proposed approach relies on two main components: a learning module and a decoding module. The former is responsible for creating multi-view representations for documents and summaries and can be used for both extractive and abstractive summarisation systems. The decoding components for extractive and abstractive summarisation are substantially different and are described in separate sections. A schematic overview of our summarisation system can be found in Figure 3.2. We call the resulting systems Correlational Extractive Summariser (CorES) and Correlational Abstractive Summariser (CorAS) respectively. In the following sections, several variants of CorES and CorAS are described and evaluated.

### 3.2.1 Learning Module

The main idea that we explore is that documents and summaries are two views of the same pieces of information. More specifically, we assume that both are views of the same latent semantics. Given that the definition of a *view* is quite broad, it is meaningful to consider two pieces of text with substantial differences as two views of the same input. Assuming that the core information contained in a document and its summary are the same, their main difference lies in their length: a summary is a restricted view of the latent information, while the document is full-length, unrestricted version of it.

Elaborating on the multi-view idea, we attempt to construct representations for documents and summaries, that will be close to the latent information. We assume that the input is described using two feature spaces,  $\mathcal{X}$  and  $\mathcal{Y}$ , with dimensionalities  $d$  and  $d'$  respectively. Without loss of generality, we use  $\mathcal{X}$  to describe documents, while  $\mathcal{Y}$  is the summary space. The objective is to learn two functions,  $u: \mathcal{X} \rightarrow \mathbb{R}^m$  and  $v: \mathcal{Y} \rightarrow \mathbb{R}^m$  that project vectors from  $\mathcal{X}$  and  $\mathcal{Y}$  to the common space  $\mathbb{R}^m$ . Furthermore, we assume that the projection of vectors from the two input spaces will also act as a dimensionality reduction step:  $m < \min(d, d')$ .

Learning the projection functions  $u$  and  $v$  can be accomplished using Canonical Correlation Analysis (see Section 2.2.2.1 for details) or Deep Canonical Correlation Analysis (see Section 2.2.2.2 for details). The overall operation of the learning process can be seen in Figure 3.3. The main feature of creating this kind of representation lies on the fact that the projected vectors for a specific document and its summary will be close in the shared space. For the experiments reported in the next sections, we mostly use DCCA with Stochastic Decorrelation Loss (refer to Section 2.2.2.3 for more details) to create the multi-view space.

## 3.3 Application on Extractive Document Summarisation

Extractive summarisation is essentially a classification problem, wherein each sentence of a document is labeled accordingly, based on whether it should be part of the summary or not. It is worth noting though, that training a classifier to act as a summariser is not always easy: the training process of a classifier relies on the availability of classification training data, which, in the case of extractive summarisation, would have to be in the form of  $(s_{ik}, l_{ik})$ , where  $s_{ik}$  refers to the  $k$ -th sentence of the  $i$ -th document and  $l_{ik}$  to the corresponding label of this sentence (0 denoting it should not be part of the summary, 1 otherwise).

Given that extractive summarisation is an artificial task and news articles or other documents

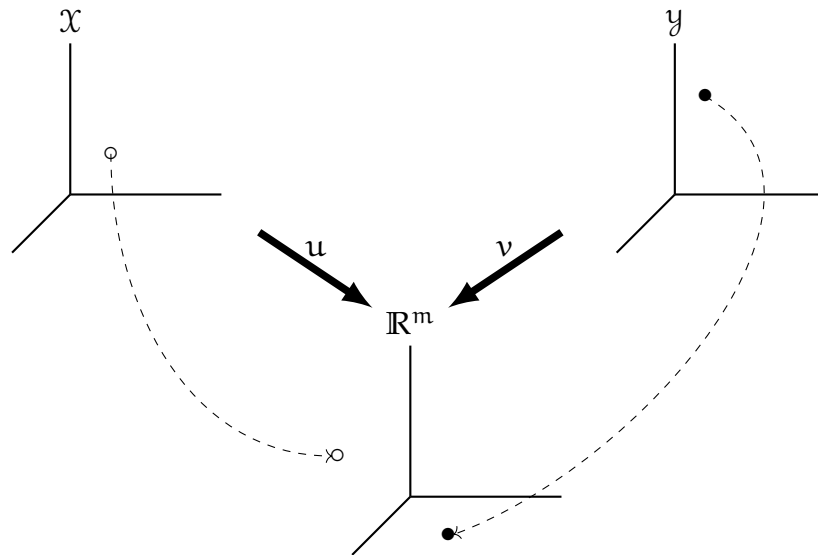


Figure 3.3: Overview of the learning module involved in multi-view document summarisation. The learning process outputs two functions,  $u$  and  $v$  that project points from  $\mathcal{X}$  and  $\mathcal{Y}$  respectively to the shared space  $\mathbb{R}^m$ .

often used do not usually come with annotations about important or summary-worthy sentences, creating such a dataset is neither an easy nor an inexpensive task. DUC datasets that were specifically designed and developed for training and evaluation of extractive summarisers are of limited usability and size.<sup>8</sup> On the other hand, datasets collected by crawling webpages, such as the CNN/DailyMail dataset (Hermann et al., 2015), contain documents paired with summaries from the original news website (human-generated, abstractive summaries).

Facing limited data availability, most works make use of the available abstractive summaries and create extractive models that aim to construct summaries similar to them. Another solution to the data problem is to automatically generate annotated data. Assigning correct labels to sentences that will be used to train a model is referred to as creating an “oracle”: a model that always knows the correct answer to the problem at hand. We experiment with both approaches: a variant of our model that includes a classifier trained on oracle data and others that simply consult the abstractive summaries.

<sup>8</sup>DUC-2001 and DUC-2002 datasets contain “extracts” (sentences from the articles that are related to a human-generated summary), but only for the multi-document summarisation task. Specifically for this dataset and this task, the training data consists of 30 document sets, each containing 10 documents; a prohibitively small size for training several types of models.



### 3.3.1 Summarisation Module

Assuming that a representation for documents and summaries has been learnt, as described in Section 3.2, each document can be represented by a vector  $\mathbf{p}$ , while each of its sentences, with a vector  $\mathbf{q}_i, i \in \{1, 2, 3, \dots, K\}$ . Then, extractive summarisation boils down to selecting  $M$  sentences to include in the summary. We distinguish and describe two ways to perform sentence selection:

- *classification*. A classifier is used to label each sentence as being part of the summary or not. Training the classifier requires the oracle annotations of the dataset. We call this variant of our model CorES-C.
- *representation-driven*, where the selection of the sentences relies only on the representations  $\mathbf{q}_i$ . This variant is called CorES-R. While this approach is quite simple, it can be seen as a case of using representations learnt in an unsupervised way to solve a downstream application.

#### 3.3.1.1 Representation-driven Selection

Representation-driven selection operates by assigning a score to each sentence, based on how similar the sentence is to the original document. A feature of the multi-view representation is that the projected vectors of a document and its summary will be close to each other. As such, we calculate a similarity score between each sentence and the document and rank the sentences accordingly.

As measures of similarity we use:

- **cosine similarity**, which is a popular measure for similarity in vector spaces. The cosine similarity between two vectors  $\mathbf{x}$  and  $\mathbf{y}$  is defined as

$$\text{sim}_{\text{cos}}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|}. \quad (3.7)$$

- the **dot product** of the two vectors:

$$\text{sim}_{\text{dot}}(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}. \quad (3.8)$$

- the **Pearson Correlation Coefficient**. Pearson correlation between two variables  $x$

and  $y$ , for which we have samples  $x_i, y_i, i = \{0, 1, 2, 3, \dots, n\}$  is defined as

$$\text{sim}_\rho(x, y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}. \quad (3.9)$$

- the **Geometric Mean of Euclidean and Sigmoid Product (GESD)**, proposed for the task of Question Answer Selection (Feng et al., 2015) as a measure of similarity that combines the L2-norm and inner product. GESD is defined as follows:

$$\text{GESD}(x, y) = \frac{1}{\|x - y\|} \cdot \frac{1}{1 + \exp(-\gamma(xy^T + c))}, \quad (3.10)$$

where  $\gamma$  and  $c$  are hyperparameters, with common values in the range from 0.5 to 1.0 for  $\gamma$  and 1.0 for  $c$ .

- the **Arithmetic Mean of Euclidean and Sigmoid Product (AESD)**, which is defined similarly to GESD:

$$\text{AESD}(x, y) = \frac{1}{\|x - y\|} + \frac{1}{1 + \exp(-\gamma(xy^T + c))}. \quad (3.11)$$

### 3.3.1.2 Selection by Classification

Multi-view models are able to produce a representation that incorporates information from all the available views. This is reflected on the objective that these models use, which prompts the maximisation of the correlation between given views; an objective that may not be enough as a loss function on its own to solve a complex problem such as document summarisation. For this reason, multi-view representations are often used as input in subsequent models, which are trained with an objective tailored to specific tasks or setups (such as, for example, cross-entropy loss in a supervised setup).

Selection by classification refers to a setup, where the multi-view representations are fed to a classifier acting as a final judge on which sentences should be included in the summary. Such an architecture learns a classifier using representations learnt by a general unsupervised model. Learning general representations in an unsupervised or semi-supervised way has been extensively studied in the past decades (Brown et al., 1992; Ando and Zhang, 2005; Mikolov et al., 2013b; Pennington et al., 2014). More recently, contextualised word embeddings, such as those produced by ELMo (Peters et al., 2018) or OpenAI GPT (Ratford et al., 2018) have

provided neural models with high quality representations, giving rise to successful models in a wide range of NLP tasks.

The effectiveness of contextualised word embeddings has been exemplified by a two step training scheme: a pre-training step usually done with a general objective, such as language modelling, followed by a downstream supervised step. This approach is used in models like BERT (Devlin et al., 2019), which have pushed the state-of-the-art in several NLP tasks. Generally speaking, our architecture also consists of an unsupervised pre-training step which learns general representations using DCCA and a supervised downstream step, implemented by the classifier. While this broadly resembles the aforementioned two-step training scheme, there is a notable difference between the two: our architecture is a pipeline of two different architectures, while BERT employs two training strategies using the same architecture.

Arguably, the representation learning step has a totally different purpose from the true objective of a classifier, which is to minimise classification errors. Moreover, the structure and operation of correlational models is such that expects only positive results. Indeed, such models do not embrace the notion of negative examples, which are of pivotal importance to the training of a classifier. This is the reason why, in order to use selection by classification, we resolve to the pipeline paradigm: the representation learning component and the classifier are trained disjointly. Finally, this setup needs annotated data, which we construct for our dataset.

### 3.3.2 Experimental Setup

**Dataset** We train and evaluate on the CNN/Daily Mail (CNN/DM) corpus, which has been compiled relatively recently to be used in the context of machine reading comprehension. This dataset consists of articles taken from the web editions of CNN and Daily Mail news websites. The two newsgroups are not equally represented in the dataset (118,497 articles from CNN vs 208,045 from DailyMail). This difference reportedly causes trained models to operate different on the two subsets of the dataset (Narayan et al., 2018a; Cheng and Lapata, 2016). The CNN/DM corpus is ideal for training and testing large-scale document summarisation techniques, since they include a set of highlights in a bullet-list format for each article.

**Alignment** The core function of an extractive summariser is deciding whether a sentence belongs to the summary or not. Hence, in our multi-view framework, space  $\mathcal{Y}$  should be a space of sentences and not a space of summaries. Conveniently, the CNN/DM dataset uses

the concept of highlights, making the space  $\mathcal{Y}$  the space of highlights. In the correlational framework though, the learning process minimises the distance between matching  $x \in \mathcal{X}$  and  $y \in \mathcal{Y}$ , while maximising it for mismatching points. Consequently, in order not to confuse our model, we would like to avoid presenting exactly the same  $x$  for multiple  $y$  points.

To this end, we transform the training data by heuristically aligning every highlight with a subset of sentences of the article. The alignment procedure is described in detail in Algorithm 3.1. In this algorithm,  $P_T$  is the set of POS tags we consult in order to calculate the match between two sentences. Our aim was to focus only on content words, that is why we consider the set  $P_T = [\text{'RB'}, \text{'RBR'}, \text{'VB'}, \text{'VBD'}, \text{'VBG'}, \text{'VBN'}, \text{'VBP'}, \text{'VBZ'}, \text{'JJ'}, \text{'JJR'}, \text{'JJS'}, \text{'NN'}, \text{'NNS'}, \text{'NNP'}, \text{'NNPS'}]$ . The heuristic described in this algorithm pairs highlights with sentences (not necessarily consecutive) that have at least some common words (vocabulary match was carried out after stemming the words). This process is done using three threshold values: if there are sentences that have a high match with a highlight sentence, they are used; otherwise, the threshold for matching sentence and highlight vocabularies is lowered. While assigning “source” sentences to corresponding highlights resembles a lot the problem of summarisation itself, it appears that a quite simple heuristic approach is adequate to generate an alignment fit for the purposes of training our model.

**Oracle** As mentioned earlier, in order to train a classifier on the large CNN/DM corpus, sentence labels are needed. A relatively inexpensive way to do that is to build an oracle: for each document, all subsets (of a specific length) of sentences are created and the one which has the best ROUGE score when compared to the abstractive summary, is kept as gold extractive summary. Since calculating oracle scores for all the possible combinations of sentences is computationally expensive, we adopt a greedy approach: we add one sentence at a time to the summary, only if it raises the ROUGE score (with respect to the gold summary). We stop adding new sentences when none of the remaining sentences improve the ROUGE score or the maximum number of sentences in the summary is reached. This approach is inspired by related work that includes converting abstractive summaries to extractive ground truth (Svore et al., 2007; Nallapati et al., 2016a; Cao et al., 2016).

**Summary Length** To generate the output summary, we select three sentences from every article. The reason for selecting three sentences is two-fold: first, in the CNN/DM corpus, each article is paired with three highlights on average. Second, three-sentence output has been used in past literature, thus making our summariser comparable with related work. For the sake of complete evaluation, though, we report ROUGE scores not only for full length summaries, but also for 275 and 75 byte summaries. All the experiments outlined in this sec-

**Algorithm 3.1** Training data alignment**Input:**

$x_i$  ▷ text of  $i$ -th article: array of sentences  
 $y_i$  ▷ highlights of  $i$ -th article: array of sentences  
 $P_T$  ▷ array of POS tags to consider for vocabulary

```

1: procedure ALIGNSET( $x_i, y_i$ )
2:   alignments is a  $\text{len}(y_i)$  size array
3:   for  $y$  in  $y_i$  do
4:     for threshold in  $[\text{len}(S_i)/2, 1, 0]$  do
5:       highlight_alignments = ALIGN( $x_i, y, \text{threshold}$ )
6:       if highlight_alignments  $\neq \emptyset$  then break
7:       alignments[i]  $\leftarrow$  highlight_alignments
8: procedure ALIGN( $x_i, y, \text{threshold}$ )
9:   highlight_vocab  $\leftarrow$  VOCABULARY( $y$ )
10:  aligned_sentences  $\leftarrow$  []
11:  for sentence in  $x_i$  do
12:    sentence_vocab  $\leftarrow$  VOCABULARY(sentence)
13:    if  $\text{len}(\text{sentence\_vocab} \cap \text{highlight\_vocab}) > \text{threshold}$  then
14:      append sentence to aligned_sentences
15:  return aligned_sentences
16: procedure VOCABULARY( $x$ )
17:  tokens  $\leftarrow$  TOKENISE( $x$ )
18:  vocab  $\leftarrow$   $\{y : \text{pos\_tag}(y) \in P_T\}$ 
19:  return vocab

```

tion were conducted in on the anonymised (entity mentions are replaced with special tokens) CNN part of the corpus, for efficiency purposes. However, ROUGE scores are calculated in non-anonymised summaries (after extracting the summary, we replace each sentence with its non-anonymised counterpart).

### 3.3.3 Experiment: Learning Representations with DCCA

This section reports experiments on our proposed extractive summariser, where the multi-view representations are learnt using the DeepCCA algorithm. We call this variant of our

summariser Deep Correlational Extractive Summariser (DCorES). We present a series of ablation experiments that thoroughly investigate each of the decisions involved in the design of DCorES.

### 3.3.3.1 Input Space and Architecture

In order to assess the importance of the original representation of the documents and the summaries, we experiment with several input spaces. Specifically, the following setups were tested:

- **ngram**: The input is ngram features from the text and the summaries. We use uni-grams and bi-grams, capping the size of the vocabulary to the 20,000 more common ones.
- **average embedding**: The input is the average word embedding of the text. We use pre-trained word2vec word embeddings,<sup>9</sup> without tuning them any further.
- **weighted embedding**: The input is a weighted sum of word embeddings of the text. Specifically, the final text embedding is calculated using *inverse document frequency* (*idf*) of each term as weight. The inverse document frequency is a measure of whether a term is common or rare among documents. For a term  $t_i$  and a set of documents  $\mathcal{D}$  that contains  $|\mathcal{D}|$  documents, *idf* is defined as:

$$\text{idf}(t_i, \mathcal{D}) = \log \frac{|\mathcal{D}|}{1 + |\{d \in \mathcal{D} : t_i \in d\}|} \quad (3.12)$$

and the weighted embedding would be calculated as

$$t = \frac{1}{n} \sum_i^n w(t_i) \text{idf}(t_i, \mathcal{D}), \quad (3.13)$$

where  $w(t_i)$  is the word embedding for  $t_i$ .

The idea of creating a representation for a text passage by calculating the *idf*-weighted sum of all its word embeddings has been used in the past, in applications ranging from question answering (Feng et al., 2015) to information retrieval (Brokos et al., 2016). We use pre-trained word2vec embedding for this variant, too.

- **encoder**: Instead of simply aggregating word embeddings, we also experiment with generating text representation using Convolutional Neural Networks (CNNs) and sequence models. We employ a hierarchical architecture wherein sentence embeddings

<sup>9</sup>Available from <https://code.google.com/archive/p/word2vec/>.

are created using convolutional encoder, and text passage embeddings are created by feeding sentence embeddings to a bidirectional Long-Short Term Memory (LSTM). The generated text passage embeddings are in turn fed to the Deep CCA module and the rest of the process is similar to that of the previous input modes. The whole network is trained with the correlation objective of Deep CCA. A schematic view of this architecture is shown in Figure 3.4. Detailed implementation information for this model can be found in Appendix A.1.

For these experiments, we fix the DCCA architecture: the lengths of its layers were set to [300,800,800,300]<sup>10</sup> on both sides and the output dimensionality to 300. Sentences for the summary are selected on a “first” basis, meaning that the model selects the first three sentences of a document that score more than the average similarity between the documents embedding and the embeddings of its sentences. Similarity is measured using cosine similarity for this study. Table 3.1 briefly outlines the results for each of the aforementioned input spaces/architectures. Despite its simplicity, ngram input provides the best results across all metrics and lengths.

	input	ROUGE-1 R	ROUGE-2 R	ROUGE-L R
<b>full length</b>	DCorES-ngram	<b>50.16</b>	<b>19.80</b>	<b>34.29</b>
	DCorES-avg_emb	42.98	15.25	29.04
	DCorES-weighted_emb	42.43	14.93	28.80
	DCorES-encoder	39.76	12.14	26.67
<b>275 bytes</b>	DCorES-ngram	<b>38.14</b>	<b>13.83</b>	<b>26.00</b>
	DCorES-avg_emb	34.80	11.66	23.90
	DCorES-weighted_emb	34.21	11.38	23.44
	DCorES-encoder	33.57	10.48	22.91
<b>75 bytes</b>	DCorES-ngram	<b>17.81</b>	<b>6.26</b>	<b>15.58</b>
	DCorES-avg_emb	16.46	5.65	14.60
	DCorES-weighted_emb	16.91	5.91	14.90
	DCorES-encoder	16.59	5.79	14.60

Table 3.1: Assessing different input setups and architectures for use with the DES summariser, on the CNN part of the dataset. ROUGE recall scores are reported for full length, 275 byte and 75 byte summaries.

<sup>10</sup>except when using the ngram input space, where the input dimensionality is equal to the vocabulary size of the ngram dictionary

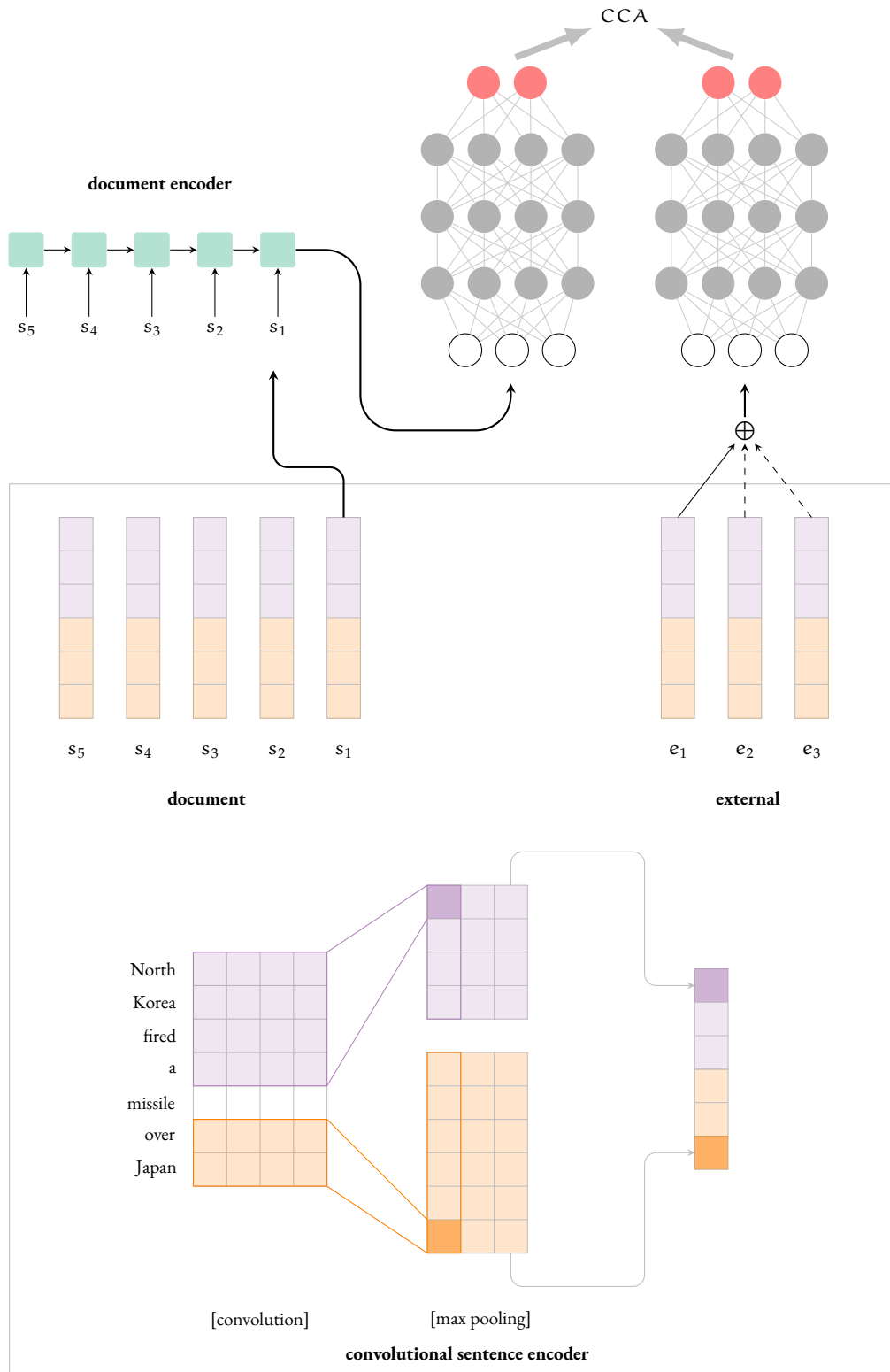


Figure 3.4: Schematic of the encoder-dcca model.



### 3.3.3.2 Is a separate classifier necessary?

As argued in Section 3.3.1.2, the idea of a classifier that explicitly groups sentences to two groups is natural for the task of extractive summarisation. In order to answer the question of whether a classifier is actually beneficial, given our architecture, we train such a classifier and compare its output to that of a summariser that does not use it.

The classifier is a multi-layer perceptron (MLP).<sup>11</sup> Specifically, it has the following characteristics:

- **features:** the classifier is trained on the output of Deep CCA with dimensionality 500. The training examples of the classifier are of the form  $(p_i, q_{ij}, l_{ij})$ , where  $p_i$  is the projected vector of the document,  $q_{ij}$  the projected vector of each document's sentence and  $l_{ij}$  the boolean label of the corresponding sentence.
- **architecture:** The first layer of the classifier calculates a bilinear form using  $p_i$  and  $q_i$  with output dimensionality 500. Subsequent layers have lengths [400, 200, 2]. The model uses ReLU activation units.
- **selection:** the three top-scoring sentences are selected as each document's summary.
- **optimisation:** is done using Stochastic Gradient Descent, with a learning rate of 0.001 and momentum 0.9.

For this experiment, we use the data of the oracle. We create negative examples to our dataset by adding  $(d_i, h_{ij})$  pairs for which highlight  $h_{ij}$  does not belong to the summary of document  $d_i$  according to the oracle. This choice is in principle counter-intuitive, since we are providing the DCCA model with multiple instances of the same  $d_i$  paired with different  $h_i$  vectors, one for every highlight, but it is one that is consistent with the classification framework. The rest of the setup of the experiment is left the same as in the *avg\_emb* run of the previous section: the input space is the ngram space, the lengths of the DCCA model layers are set to [300, 800, 800, 300] and the output dimensionality to 300. We conduct this experiment using the *avg\_emb* setup for computational efficiency.

The results of this experiment are shown in Table 3.2, where DCorES-C refers to the classification setup, and DCorES-R to the selection by representation setup. Surprisingly, the classification setup lacks behind the selection by representation one. One reason for this could be the aforementioned counter-intuitive use of the DCCA model.

---

<sup>11</sup>The model was implemented in PyTorch,<sup>12</sup> version 0.2.0. .

	mode	ROUGE-1 R	ROUGE-2 R	ROUGE-L R
<b>full length</b>	DCorES-R	<b>42.98</b>	<b>15.25</b>	<b>29.04</b>
	DCorES-C	39.94	10.94	30.53
<b>275 bytes</b>	DCorES-R	<b>34.80</b>	<b>11.66</b>	<b>23.90</b>
	DCorES-C	27.34	6.47	21.55
<b>75 bytes</b>	DCorES-R	<b>16.46</b>	<b>5.65</b>	<b>14.60</b>
	DCorES-C	12.12	2.18	10.47

Table 3.2: Comparing selection by classification and selection by representation, on the CNN part of the dataset. ROUGE recall scores are reported for full length, 275 byte and 75 byte summaries. DCorES-R refers to selection by representation, while DCorES-C to the classification setup.

### 3.3.3.3 Choosing a metric

By now, we have established that the most successful model is the one that uses ngram input representation and selects sentences solely by using the representations. The present section focuses on the choice of cosine similarity as a similarity metric to perform selection by representation.

To this end, we experiment with using different similarity metrics (dot product, euclidean distance, pearson correlation, GESD, AESD). Specifically, we test the five metrics in two different setups:

- *first*, where the model selects the first three sentences of a document that score more than the average similarity between the the document’s projected embedding and the projected embedding of its sentences. As such, these runs use not only the similarity of the sentences to the document, but they also indirectly use the order of sentences in the document.
- *best*, where the model selects the three top-scoring sentences of the document. This section of the table investigates whether similarity in the shared space alone is a good way for selecting sentences for the final summary.

In general, it seems that the *first* setup (choosing the first sentences over a threshold) results to significantly better results. It would be fair to conclude that the representation alone is not enough to rank the sentences according to their usefulness for a summary; instead, a second bit of information, the order of the sentences in the document, can boost the results.

	system	ROUGE-1 R	ROUGE-2 R	ROUGE-L R		
full length	FIRST	DCorES-R-first-cosine	50.16	19.80	34.29	
		DCorES-R-first-gesd	48.30	18.92	32.94	
		DCorES-R-first-aesd	48.00	18.73	32.69	
		DCorES-R-first-dot	<b>50.28</b>	<b>19.87</b>	<b>34.44</b>	
		DCorES-R-first-eucl	48.12	18.88	33.05	
		DCorES-R-first-corr	50.19	19.78	34.28	
		BEST	DCorES-R-best-cosine	44.90	15.09	29.53
	DCorES-R-best-gesd		42.62	14.23	28.20	
	DCorES-R-best-aesd		42.70	14.22	28.22	
	DCorES-R-best-dot		44.87	15.04	29.40	
	DCorES-R-best-eucl		44.87	15.04	29.40	
	DCorES-R-best-corr		44.98	<i>15.16</i>	<i>29.60</i>	
	275 bytes		FIRST	DCorES-R-first-cosine	<b>38.14</b>	13.83
		DCorES-R-first-gesd		37.15	13.46	25.37
DCorES-R-first-aesd		37.03		13.36	25.22	
DCorES-R-first-dot		38.11		<b>13.86</b>	25.98	
DCorES-R-first-eucl		37.05		13.35	25.39	
DCorES-R-first-corr		38.12		13.82	25.97	
BEST		DCorES-R-best-cosine		33.63	10.16	22.41
		DCorES-R-best-gesd	32.22	9.54	21.56	
		DCorES-R-best-aesd	32.26	9.52	21.57	
		DCorES-R-best-dot	33.58	<i>10.19</i>	22.48	
		DCorES-R-best-eucl	31.91	9.49	21.36	
		DCorES-R-best-corr	33.68	10.16	22.44	
		75 bytes	FIRST	DCorES-R-first-cosine	<b>17.81</b>	<b>6.26</b>
DCorES-R-first-gesd				17.38	6.19	15.28
DCorES-R-first-aesd	17.28			6.17	15.19	
DCorES-R-first-dot	17.80			6.32	15.58	
DCorES-R-first-eucl	17.42			6.04	15.25	
DCorES-R-first-corr	17.79			6.25	15.56	
BEST	DCorES-R-best-cosine			16.27	4.35	14.03
	DCorES-R-best-gesd		15.28	3.77	13.12	
	DCorES-R-best-aesd		15.27	3.76	13.12	
	DCorES-R-best-dot		16.06	4.17	13.87	
	DCorES-R-best-eucl		15.15	3.75	13.02	
	DCorES-R-best-corr		16.34	<i>4.37</i>	<i>14.09</i>	

Table 3.3: Impact of different metrics on the performance of DCorES-R. For details on the metrics, refer to Section 3.3.1.1. Scores on italics are the best for each length and mode (top/-first) group, while scores in boldface are the best scores for each length group.

In terms of the metrics themselves, while changing metrics within the same setup leads to consistent results, evidently, cosine similarity gives best results across all lengths.

#### 3.3.3.4 Choosing a threshold

All systems that employ the *first* selection setup pick sentences that have a score larger than the per document average similarity. To investigate whether this decision holds merit, we experiment with the use of a fixed threshold for the whole dataset, testing threshold values in the range  $[0, 0.3]$  with a step of 0.002. For this experiment, we use the correlation metric.

The performance of the resulting systems is plotted in Figure 3.5, where the x axis represents the threshold value and the y axis ROUGE-1 recall values. The maximum value is 50.18 (for a threshold value of 0.156). This score does not differ significantly from the scores we get from using the per document average score, considering that we get a score of 50.19 using the correlation metric.

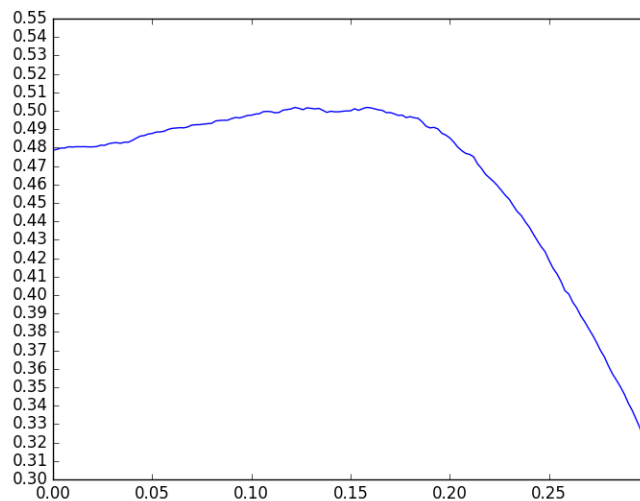


Figure 3.5: ROUGE-1 recall values for different values of threshold in  $[0, 0.3]$  with step of 0.002. The evaluation was carried out using the correlation metric. The maximum value is 50.18.

#### 3.3.3.5 Choosing output dimensionality

Most representation learning algorithms introduce one extra hyperparameter to the learning problem: the output dimensionality. In order to explore the effect of different output dimensionalities on the performance of DCorES-R, we freeze the rest of our architecture and change the length of the output: the network's lengths are set to  $[300, 800, 800, \text{out}]$ ,

	dim	ROUGE-1 R	ROUGE-2 R	ROUGE-L R
<b>full length</b>	100	48.17	18.55	32.80
	500	<b>49.34</b>	<b>19.52</b>	<b>33.83</b>
	1000	49.03	19.15	33.55
<b>275 bytes</b>	100	37.61	13.56	25.60
	500	<b>37.98</b>	<b>13.88</b>	<b>25.99</b>
	1000	37.83	13.82	25.93
<b>75 bytes</b>	100	17.74	6.24	15.60
	500	<b>18.20</b>	<b>6.61</b>	<b>16.02</b>
	1000	17.86	6.16	15.63

Table 3.4: Impact of different output dimensionalities on the performance of DCorES-R.

sentences are selected by using the "first" setup and cosine similarity is used as the similarity metric. The corresponding results are shown in Table 3.4. It can be seen that a dimensionality of 100 is quite low and does not result to informative representations, while a dimensionality of 1000 is quite large and we start seeing a degradation in performance.

### 3.3.3.6 Choosing views

DeepCCA and DCCA with Stochastic Decorrelation Loss are proposed for two-view setups, but are fairly straightforward to extend to work for setups with more than two views. For the DCorES system, this can be achieved by modifying the objective and the sentence scoring mechanism. Assuming more than two views, during training, the objective should account for the correlation between all pairs of available views. Moreover, during testing, scores for each sentence can be calculated as the sum of the scores comparing the document, the summary and any other view.

Specifically, in a three-view setup, where we have multi-view representations  $v_1, v_2$  and  $v_3$  for the document, a sentence from the document and a third view respectively, the score can be calculated as

$$\text{score} = \text{sim}(v_1, v_2) + \text{sim}(v_3, v_2). \quad (3.14)$$

In the present section, we experiment with using information from several views of the dataset:

- *article title*: one could think of the title of an article as a one-sentence summary of it. The rationale for using it as a view for summarisation is that the title, the summary and the document are three different versions of the article with different length limitations, with each one being the summary of the other.
- *related questions*: The recently proposed dataset NewsQA (Trischler et al., 2017) is a question answering dataset constructed by generating questions for CNN articles. The articles were selected by randomly choosing 12,744 articles from the 90,266 CNN articles of the CNN/DM dataset. Crowdworkers were presented with the title and the summary of an article and were instructed to formulate questions on the article. One could regard those questions as a separate view of the news articles, treating them as another source of related information. In order to run experiments utilising this extra view, we extracted the subset of the questions that can actually be answered by the article text.<sup>13</sup>
- *image captions*: the captions of images that accompany articles can be a good source of information for summarisers. As shown in the work of Narayan et al. (2018a), captions and title can aid a summarisation system in producing better quality summaries.

We fix the rest of the details of the setup across all runs: the output dimensionality is set to 300 and sentences are selected by using the *first* setup using cosine similarity. Scoring works as described in Equation 3.14. Brief experimentation with weighting different views differently did not result in significant gains, although clearly, some views have larger impact than others.

Table 3.5 presents full length results for different types of views. The runs annotated with  $tr_{a,b}$  use information from views  $a$  and  $b$  during training, while the runs  $t_{a,b}$  consider the projected vectors of views  $a$  and  $b$  during test time. For example, the run DCorES-R-first+ $tr_q$  is trained on text, summary and question views, but on test time, the question view is not considered (we suppose that we have no available questions for the articles of the test set). Similarly, DCorES-R-first+ $tr_q+t_q$  considers questions during both training and test time. The essential views (text and summary) are not mentioned in the Table, since they are common in all runs.

In general, the results do not suggest any significant gain by using more than two views. However, it seems that the title view adds some more information than other views, resulting in slightly better results. Interestingly, we do not observe any loss in performance as we add

---

<sup>13</sup>This is not the case for all the questions, since some of them cannot be answered by reading the article text.

more views.

**Missing views** The runs which consider a specific view during training, but not while testing can be referred to as *missing view* setups. Such setups are a common application of multi-view representation learning techniques (Xu et al., 2015a), where the multi-view representations are able to partly recover missing information in such setups. Specifically, since part of learning the representation relies on creating correlated projections, projecting one view results in a representation that is close to the one that would be created if all input representations were available. This behaviour is particularly useful in situations with limited data, for example where data from one view are expensive to obtain for the whole dataset, are corrupted, or simply not present (they were never gathered).

Interestingly, in our experiments, missing view runs have results really close to the ones that consider the corresponding views at test time. Such a result is evidence for the quality of the generated representations, in the sense that the vectors produced in missing view setups are close to or have similar effect to the ones that would be generated if all views are considered.

### 3.3.3.7 Discussion

In this section, we compare the best performing variant of DCorES with several extractive summarisation systems. The comparison, which can be seen in Table 3.6 takes into account the following models:

- **LEAD-3** is the LEAD baseline that selects the first three sentences of every document.
- **Deep-Cls** is the best scoring system of the work of Nallapati et al. (2016a). It uses Recurrent Neural Networks (RNNs) and takes into account the order of the sentences in the original document.
- **NN-SE** is the best scoring system of the work of Cheng and Lapata (2016), which is a neural sentence extraction model.
- **SideNet** is the best performing system of the work of Narayan et al. (2018a), a neural model which considers side information, such as title and image captions apart from the document text and summary.

It becomes clear from this comparison, that our proposed system lacks significantly behind state-of-the-art summarisers. However, it can be seen that it is able to outperform the admittedly not-easy-to-beat LEAD baseline just by learning a good representation for summaries and documents.

	dim	ROUGE-1 R	ROUGE-2 R	ROUGE-L R
<b>three views</b>	DCorES-R-first+tr <sub>q</sub>	48.10	18.72	32.58
	DCorES-R-first+tr <sub>q</sub> +t <sub>q</sub>	47.55	18.18	32.19
	DCorES-R-first+tr <sub>t</sub>	48.44	18.47	32.76
	DCorES-R-first+tr <sub>t</sub> +t <sub>t</sub>	<b>48.56</b>	<b>18.77</b>	<b>32.91</b>
	DCorES-R-first+tr <sub>c</sub>	48.10	18.72	32.58
	DCorES-R-first+tr <sub>c</sub> +t <sub>c</sub>	47.91	18.51	32.63
<b>four views</b>	DCorES-R-first+tr <sub>t,c</sub>	<i>48.10</i>	<i>18.72</i>	<i>32.58</i>
	DCorES-R-first+tr <sub>t,c</sub> +t <sub>t</sub>	47.36	18.09	32.29
	DCorES-R-first+tr <sub>t,c</sub> +t <sub>c</sub>	47.36	18.09	32.29
	DCorES-R-first+tr <sub>t,c</sub> +t <sub>t,c</sub>	47.36	18.09	32.29
	DCorES-R-first+tr <sub>q,t</sub>	47.36	18.09	32.29
	DCorES-R-first+tr <sub>q,t</sub> +t <sub>t</sub>	47.36	18.09	32.29
	DCorES-R-first+tr <sub>q,t</sub> +t <sub>q</sub>	47.47	18.23	32.51
	DCorES-R-first+tr <sub>q,t</sub> +t <sub>q,t</sub>	47.47	18.23	32.51
<b>five views</b>	DCorES-R-first+tr <sub>q,t,c</sub>	47.28	18.00	31.87
	DCorES-R-first+tr <sub>q,t,c</sub> +t <sub>t</sub>	47.45	18.26	32.24
	DCorES-R-first+tr <sub>q,t,c</sub> +t <sub>q</sub>	47.20	18.00	31.86
	DCorES-R-first+tr <sub>q,t,c</sub> +t <sub>q,t</sub>	47.45	18.26	32.24
	DCorES-R-first+tr <sub>q,t,c</sub> +t <sub>t,c</sub>	47.20	18.00	31.86
	DCorES-R-first+tr <sub>q,t,c</sub> +t <sub>q,c</sub>	<i>47.54</i>	<i>18.41</i>	<i>32.30</i>
	DCorES-R-first+tr <sub>q,t,c</sub> +t <sub>q,t,c</sub>	47.20	18.00	31.86

Table 3.5: Full length ROUGE results on using different views. In subscripts, the t stands for title, q for questions and c for captions (of photos). Scores on italics are the best for each group of views (three, four, five views), while scores in boldface are the best across all runs.

### 3.3.4 Experiment: Learning Representations with CCA

We also explore the effect of a CCA training algorithm on the task of extractive summarisation.

#### 3.3.4.1 Input Space

We experiment with two different types of features for the input data:

- *ngram features*: we calculate word bigrams for all article texts and highlights.



	system	ROUGE-1 R	ROUGE-2 R	ROUGE-L R
full length	LEAD-3	47.57	18.79	32.24
	NN-SE	51.70	19.70	45.70
	SideNet	<b>54.20</b>	<b>21.60</b>	<b>48.10</b>
	DCorES-R-first-dot	50.28	19.87	34.44
275 bytes	LEAD-3	37.07	13.53	25.21
	NN-SE	38.60	13.90	34.3
	SideNet	<b>39.70</b>	<b>14.70</b>	<b>35.20</b>
	DCorES-R-first-dot	38.11	13.86	25.98
75 bytes	LEAD-3	<b>21.90</b>	<b>7.20</b>	11.60
	NN-SE	20.30	<b>7.20</b>	14.80
	SideNet	20.20	7.10	14.60
	DCorES-R-first-dot	17.80	6.32	<b>15.58</b>

Table 3.6: Comparing the best performing variant of DCorES (DCorES-R-first-dot) with other summarisers on the CNN part of the test corpus.

- *masked ngrams*: before calculating word bigram features, we mask the input by replacing every token with a tuple  $(p, b)$ , where  $p$  is the part of speech tag and  $b$  is the Brown cluster the token belongs to. To this end, we use pre-calculated Brown clusters from the work of [Stratos et al. \(2014\)](#). Masking is aimed to be performed on content words; thus, tokens that are annotated as conjunctions, determiners, modals, pronouns, interjections and wh-words, are not masked. One can think of this masking as a first step in lowering the dimension of the input space: the space of the bigrams generated from the masked versions is substantially lower-dimensional.

Results for different input spaces can be seen in Table 3.7 for the DailyMail part of the dataset. In general, masked variants of our model seem to perform better than the ones operating on ngrams. This justifies our choice to mask the input with part of speech tags and Brown clusters.

### 3.3.4.2 Selection Setup

As in previous experiments, we select three sentences from every article to generate a summary. For the selection process, we explore two different approaches, the “first” (select the first sentences of the document that have a similarity score larger than a predefined threshold

		DailyMail		
	system	ROUGE-1 R	ROUGE-2 R	ROUGE-L R
<b>full length</b>	CorES-first-ngrams	55.46	<b>24.32</b>	34.68
	CorES-best-ngrams	56.55	22.67	35.74
	CorES-first-masked	55.02	24.13	34.42
	CorES-best-masked	<b>57.02</b>	22.92	<b>36.05</b>
<b>275 bytes</b>	CorES-first-ngrams	40.99	<b>16.45</b>	25.84
	CorES-best-ngrams	37.10	13.93	24.40
	CorES-first-masked	<b>41.09</b>	16.60	<b>25.94</b>
	CorES-best-masked	36.42	13.73	24.00
<b>75 bytes</b>	CorES-first-ngrams	22.35	8.46	19.80
	CorES-best-ngrams	22.02	<b>8.70</b>	19.63
	CorES-first-masked	<b>22.62</b>	8.66	<b>20.06</b>
	CorES-best-masked	21.62	8.57	19.29

Table 3.7: Ablation for different inputs and selection setups, on the DailyMail part of the test corpus.

value) and “best” (top scoring sentences). At places where those constraints fail to generate a three-sentence summary, one with less than three sentences is given as output.

Results for different selection setups can be seen in Table 3.7 for the DailyMail part of the dataset. In this case, the “best” variants seem to perform better, suggesting that the cosine similarity is indeed an indicator for informative sentences for summarisation.

### 3.3.4.3 Discussion

In this section, we compare the two best performing variants of CorES with several extractive summarisation systems. We present a comparison only on the DailyMail part of the dataset, since this is what most works evaluate on. The comparison, which can be seen in Table 3.8 takes into account the following models:

- **LEAD-3** is the LEAD baseline that selects the first three sentences of every document.
- **Deep-Cls** is the best scoring system of the work of Nallapati et al. (2016a). It uses Recurrent Neural Networks (RNNs) and takes into account the order of the sentences in the original document.

DailyMail				
	system	ROUGE-1 R	ROUGE-2 R	ROUGE-L R
full length	LEAD-3	47.57	18.79	32.24
	NN-SE	56.00	<b>24.90</b>	<b>50.20</b>
	CorES-best-ngrams	<b>56.55</b>	22.67	35.74
	CorES-first-masked	55.02	24.13	34.42
275 bytes	LEAD-3	37.07	13.53	25.21
	Deep-Cls	<b>42.20</b>	16.80	<b>35.00</b>
	NN-SE	<b>42.20</b>	<b>17.30</b>	34.80
	CorES-best-ngrams	37.10	13.93	24.40
	CorES-first-masked	41.09	16.60	25.94
75 bytes	LEAD-3	21.90	7.20	11.60
	Deep-Cls	<b>26.20</b>	<b>10.70</b>	14.40
	CorES-best-ngrams	22.02	8.70	19.63
	CorES-first-masked	22.62	8.66	<b>20.06</b>

Table 3.8: Comparing the best performing variants of CorES (CorES-best-ngrams and CorES-first-masked) with other summarisers on the DailyMail part of the test corpus.

- **NN-SE** is the best scoring system of the work of [Cheng and Lapata \(2016\)](#), which is a neural sentence extraction model.
- **SideNet** is the best performing system of the work of [Narayan et al. \(2018a\)](#); a neural model which considers side information, such as title and image captions apart from the document text and summary.

It can be seen that, using CCA to generate representations for the task of extractive summarisation, leads to a system that can outperform the LEAD baseline, as is the case with our DCorES summariser. However, our system is not able to outperform state-of-the-art summarisation systems.

Interestingly, the performance of CorES is consistently higher than that of our DCorES system. While this may be counter-intuitive given the improved capabilities Deep CCA comes with, it seems, that for the task of extractive summarisation the way we formulate it, the linear transformations of CCA work better.

## 3.4 Application on Abstractive Document Summarisation

Our multi-view framework includes a learning module that can be of use in both extractive and abstractive summarisation. In order to expand our framework to accommodate the task of abstractive summarisation, we need a text generation mechanism. We call the abstractive summariser consisting of the learning module and the decoder described in the next section Correlational Abstractive Summariser (CorAS).

### 3.4.1 Decoding Module

For the task of abstractive summarisation, we propose a decoder based on a Markov Chain Monte Carlo (MCMC) sampler. The decoder is expected to output a highlight  $y$  given an article  $x$ , and the projection functions  $u$  and  $v$  learnt from the CCA training. Its main goal is to output sentences whose projected vector  $v(y)$  will be as close to the projected vector of the document  $u(x)$  as possible in the shared space.

The output space  $\mathcal{Y} \subseteq \Lambda^*$  of the decoder is the set of strings over some alphabet  $\Lambda$ . For example,  $\mathcal{Y}$  could be the set of all  $n$ -gram chains possible over some  $n$ -gram set or the set of possible composition of atomic phrases, similar to phrase tables in phrase-based machine translation (Koehn et al., 2007). Our proposed approximate decoding algorithm makes use of such a phrase table  $\mathcal{P}$  such that every  $y$  can be decomposed into a sequence of consecutive phrases  $p_1, p_2, \dots, p_r \in \mathcal{P}$ .

We implement a Metropolis-Hastings (MH) algorithm that assumes the existence of a black-box sampler  $q(y|y')$  – the proposal distribution. The sampler randomly chooses two endpoints  $k$  and  $l$  in  $y'$  and, if possible, replaces all the words in between these two words ( $y'_k, \dots, y'_l$ ) with a phrase  $p$  such that in the training data, there is an occurrence of the new phrase  $p$  after the word  $y'_{k-1}$  and before the word  $y'_{l+1}$ . As such, we are required to create a probabilistic table of the form  $Q : \Lambda \times \mathcal{P} \times \Lambda \rightarrow \mathbb{R}$  that maps a pair of words  $w, w' \in \Lambda$  and a phrase  $p \in \mathcal{P}$  to the probability  $Q(p|w, w')$ . We construct such a phrase table by scanning the training part of the corpus and using relative frequency count to estimate  $Q$ : we count the number of times each phrase  $p$  appears between the context words  $y$  and  $y'$  and normalise.

Since we are interested in maximising the cosine similarity between  $v(y)$  and  $u(x)$ , after each sampling step, we check whether the cosine similarity of the new  $y$  is higher (regardless of whether it is being accepted or rejected by the MH algorithm) than that of any  $y$  so far. We return the best  $y$  sampled. The “true” unnormalised distribution we use in the accept-

rejection step is the exponentiated value of the cosine similarity between  $\mathbf{u}(\mathbf{x})$  and  $\mathbf{v}(\mathbf{y})$ . This means that for a given  $\mathbf{x}$ , the MH algorithm implicitly samples from the following distribution  $P$ :

$$P(\mathbf{y}|\mathbf{x}) = \frac{\exp\left(\frac{\langle \mathbf{u}(\mathbf{x}), \mathbf{v}(\mathbf{y}) \rangle}{\|\mathbf{u}(\mathbf{x})\| \cdot \|\mathbf{v}(\mathbf{y})\|}\right)}{Z(\mathbf{x})}, \quad (3.15)$$

where

$$Z(\mathbf{x}) = \sum_{\mathbf{y}' \in \mathcal{Y}} \exp\left(\frac{\langle \mathbf{u}(\mathbf{x}), \mathbf{v}(\mathbf{y}') \rangle}{\|\mathbf{u}(\mathbf{x})\| \cdot \|\mathbf{v}(\mathbf{y}')\|}\right). \quad (3.16)$$

The probability distribution  $P$  has a strong relationship to the von Mises-Fisher distribution, which is defined over vectors of unit vector. The von Mises-Fisher distribution has a parametric density function  $f(\mathbf{z}; \boldsymbol{\mu})$  which is proportional to the exponentiated dot product between the unit vector  $\mathbf{z}$  and some other unit vector  $\boldsymbol{\mu}$  which serves as the parameter for the distribution. The main difference between the von Mises-Fisher distribution and the distribution defined in Equation 3.15 is that we do not allow *any* unit vector to be used as  $\frac{\mathbf{v}(\mathbf{y})}{\|\mathbf{v}(\mathbf{y})\|}$ ; only those which originate in some output structure  $\mathbf{y}$ . As such, the distribution in Equation 3.15 is a re-normalised version of the von-Mises distribution, after elements from its support are removed.

**Controlling for length** The decoder in its vanilla form tends to generate small sentences. For this reason, we modify it so that it maximises a score for each sentence, which is not only based on the cosine similarity between the projected vectors, but also takes into account also the length of the generated sentence. Specifically, the score for a given sentence  $s$  is calculated as follows:

$$\text{score}(s) = \text{sim}_{\text{cos}} + \lambda(\min(L_C, |s|) - (|s| - L_C)) \quad (3.17)$$

where  $|s|$  is the sentence length (number of tokens),  $\lambda$  a parameter which we tune over the validation set and  $L_C$  a maximum length which we set to 20.  $L_C$  serves as a cap in length, so that the sentence scores do not keep increasing uncontrollably with the sentence size.

**Simulated Annealing** Since we are not interested in sampling from the distribution  $P(\mathbf{y}|\mathbf{x})$ , but actually find its mode, we use simulated annealing with our MH sampler. This means that we exponentiate by a  $\frac{1}{t}$  term the unnormalised distribution we sample from, and decrease this temperature  $t$  as the sampler advances. We start with a temperature  $T = 10,000$ , and multiply  $t$  by  $\tau = 0.995$  at each step. As such, the sampler starts with an exploratory phase where it jumps from different parts of the search space to others, and as the temperature decreases, it makes smaller jumps.

The algorithm of our proposed decoder is shown in Algorithm 3.2. A variant of this decoder is also used in our work on image captioning (see Chapter 4).

---

**Algorithm 3.2** MCMC Decoder for Abstractive Summarisation
 

---

**Input:**

$x$	▷ an input example
$\rho$	▷ a similarity metric
$u, v$	▷ two projection functions
$Q$	▷ a probabilistic phrase table
$\eta \geq 0$	▷ a constant
$\tau \in (0, 1)$	▷ a constant
$T$	▷ a starting temperature
score	▷ the scoring function of Equation 3.17

1: **procedure** CCA\_DECODER

2:   Let  $y^*$  be an arbitrary point in the output space.

3:    $y' \leftarrow y^*$ .

4:    $t \leftarrow T$ .

5:   **while** temperature  $t$  is below a given value **do**

6:     uniformly choose two different integers  $i$  and  $j$ , between 1 and  $|y'|$

7:     choose randomly a phrase  $p$  from  $Q(p \mid y'_{i-1}, y'_{j+1})$

8:      $y \leftarrow y'_1 \cdots y'_{i-1} p y'_{j+1} \cdots y'_{|y'|}$

9:     **if**  $\text{score}_s(y) \geq \text{score}_s(y^*)$  **then**

10:        $y^* \leftarrow y$

11:        $\alpha_0 \leftarrow \frac{\exp\left(\frac{1}{t} \text{score}(y)\right)}{\exp\left(\frac{1}{t} \text{score}(y')\right)}$

12:        $\alpha_1 \leftarrow \frac{|y|^2 Q(y'_i \cdots y'_j \mid y'_{i-1}, y'_{j+1})}{|y'|^2 Q(y_i \cdots y_j \mid y_{i-1}, y_{j+1})}$

13:        $\alpha \leftarrow \{1, \alpha_0 \cdot \alpha_1\}$

14:       uniformly sample a number  $k$  from  $[0, 1]$

15:       **if**  $k < \alpha$  **then**

16:          $y' \leftarrow y$

17:        $t \leftarrow \tau t$

**return**  $y^*$

---

### 3.4.2 Experimental Setup

For each article, we generate three highlights. We experimented with different setups:

- decoding by using the vector of the whole article.
- decoding by using the vectors of the first sentences.
- decoding by using the vector resulting from adding the whole text vector to the vector of each sentence.

**Masked phrase table and copying** Usually, summaries of document contain terms and phrases that are not frequent in the general sense and could not be easily predicted by a language model. This happens mainly because there are parts of the summary, such as entity mentions or specific action mentions that may not be common or existent at all in the training set. In order for a summariser to be able to construct more meaningful summaries, a copying mechanism that can copy parts of the original document to the summary is required. In our model, the masking step described in Section 3.3.4.1 provides a way to deal with copying in a very shallow way.

We construct a phrase table with n-gram chains of masked tokens. Consequently, the sampler instead of generating normal, readable sentences, outputs masked sentences. We further process the output of the sampler by substituting each masked token with a token from the original document that has the same mask. In cases where more than one tokens with the same mask exist, the first one (by order of appearance on the document) is selected. It is worth noting that this copying mechanism is by no means context-aware. However, it seems that the Brown cluster part of the mask operates as a good criterion on selecting useful, summary-worthy tokens for copying. Tokens with masks that are not present in the original document are substituted by common tokens with the same mask.

### 3.4.3 Discussion

We use ROUGE score to assess informativeness and fluency of the generated summaries. Contrary to extractive summarisation, for this task, precision and F scores are important, as the summaries are not just selected from the original document, but generated. Our best-performing setup is the one that generates one highlight from each of the first three sentences, which achieved a 31.01 ROUGE-1 F1 score, whereas the state-of-the-art model (See et al., 2017) was able to achieve 39.53 in the same dataset. Results for CorAS are not encouraging, suggesting that substantial improvements should be made in the generation part.

A small benchmark carried out to evaluate the unmasking process, showed that given the correct masked output, a 80% of the original highlight can be retrieved. This leads us to believe that the generation part has to be improved and that the unmasking procedure is not the main reason for the low scores.

## **3.5 Conclusions**

In this chapter, we present extractive and abstractive summarisation models aiming to generate and use good representations for news texts and summaries. This summarisation work is motivated by the idea that using multi-view representations could result in good quality automatic summaries. Extensive experimentation on a large-scale dataset using the techniques of Canonical Correlation Analysis and Deep Canonical Correlation Analysis, exploiting more than two views and using several setups, has shown, that the described models are not enough to produce summaries whose quality can outperform state-of-the-art models.

However, the fact that, by using CCA or Deep CCA alone, a system can outperform the relatively hard-to-beat LEAD baseline, highlights the quality of the learnt representations. The effectiveness of complementing state-of-the-art models with such representations, thus getting the best of both worlds, is yet to be fully explored. Moreover, it would be interesting to test the applicability of multi-view document and summary representations in non-artificial missing data setups, where missing views are an actual issue, either due to availability or due to computational efficiency restrictions.





## Chapter 4

# Canonical Correlation Inference for Image Captioning

*... humans can understand a real-world scene quickly and accurately, saccading many times per second while scanning a complex scene. Each of these glances carries considerable information.*

*(Fei-Fei et al., 2007)*

Image understanding, a field of interest of Computer Vision (CV), includes techniques and methods that allow computational models to exhibit intelligent behaviour in regards to images and visual content. The CV community has introduced a number of tasks in order to encourage and monitor the performance of vision systems, the most fundamentally challenging of which are, probably, the tasks of object recognition and object detection (Szeliski, 2011).

In order to get a better understanding of the goals of those tasks, an example from the MSCOCO object recognition dataset (Lin et al., 2014) is shown in Figure 4.1. Object recognition refers to classifying an image as containing a specific object from a pre-defined set of objects. Given that real-life photographs may depict complex scenes, this is naturally a multi-way classification problem: the image of the example contains seven different objects from the pre-defined set. Object detection is the task of localising an object in an image and subsequently drawing a bounding box around it. In the example image, such a box is drawn around all objects (person, bottle, doughnut, chair, dining table, teddy bear) identifying their limits.

A more complex task that includes a text component is that of Visual Question Answering (VQA; see Wu et al. 2017b for a survey). In VQA, the goal is to give accurate responses to



1. a boy seated at the table with stuff teddy bear drinking milk and eating a doughnut.
2. a little boy is drinking chocolate milk with his teddybear.
3. a young boy drinking chocolate milk with his teddy bear.
4. a small boy with a drink a donut and a teddy bear
5. a child drinking chocolate milk and eating a donut

Figure 4.1: An example photo from the MS-COCO dataset, along with annotations and captions. The right side shows the photo from the left, but with visible annotations on objects. Each of the objects (person, bottle, doughnut, chair, dining table, teddy bear) has a bounding box drawn around it. This figure showcases three tasks pertaining to images: object detection (finding the bounding boxes), object recognition (classifying the objects in one of a number of categories) and image captioning.

questions regarding an image. An example from the VQA dataset (Antol et al., 2015) can be seen in Figure 4.2. Despite criticism claiming that systems can perform adequately without even having to consult the images on this dataset (Agrawal et al., 2016), it is apparent that a balanced and fair version of the task would require a certain level of image understanding and inference over the processed image.

Although hard to measure, a person’s perception of a scene is usually judged by their reported description of it, in both academic and real-life setups. Indeed, this is in part a reason for the heavy focus on automatic image description by both researchers and practitioners. Owing to advances in CV and NLP, image captioning has been a point of convergence of the two communities and a testbed for multimodal applications.

In this chapter, we extend our exploration of multi-view learning to the bimodal problem of image description generation. More specifically, we investigate the idea of using Canonical



Figure 4.2: Example photos from the VQA dataset (Antol et al., 2015) and corresponding questions.

Correlation Analysis to represent images and their descriptions and propose an algorithm to create novel captions for images.

## 4.1 Image Captioning

Automatic image captioning or automatic image description generation is the task of automatically assigning textual descriptions to images. A model able to successfully generate image descriptions gives the impression of visual and contextual understanding, although understanding is not the only prerequisite for description (Bernardi et al., 2016). Despite the recent success of image captioning systems, captioning is admittedly far from being solved. A captioning system should be able to overcome several difficulties, since image descriptions are expected to be accurate, but at the same time structured, coherent and well-formed sentences or phrases. The linguistic constraint of well-formedness, which is not trivial to achieve while maintaining high accuracy, is what calls for joint efforts of the CV and NLP communities in this task.

Image captioning has a number of applications, including image retrieval. The task of identifying one or more images fulfilling an informational need is becoming increasingly harder as the amount of images people produce and interact with increases. Successful image description generators can create a middle ground between the complex domains of images and users, by providing a mapping from the former to text, where information retrieval techniques have been proven to give adequate results. Moreover, image descriptions or content-based image keywords and phrases can aid in the construction of interfaces for visually impaired people.

Image description generation has been thoroughly studied in various setups and variances.

In an effort to simplify the task and not account for the complexity of natural language generation systems, early approaches cast image description as a retrieval task. As such, the problem is reduced to learning a similarity metric between images and for every new, unseen image, a set of similar images is retrieved and generation proceeds using the descriptions of those images. Queries for similar images can be posed against a visual space (Ordenez et al., 2011; Mason and Charniak, 2014) or a multimodal space, where images and descriptions have been projected (Farhadi et al., 2010; Hodosh et al., 2013).

Non-retrieval approaches to image description generation have for a long time relied on a set of predefined sentence templates (Kulkarni et al., 2011; Elliott and Keller, 2013; Yang et al., 2011) or used syntactic trees (Mitchell et al., 2012), while more recently, methods that use neural models (examples include Kiros et al. 2014; Vinyals et al. 2015) have appeared, that avoid the use of any kind of predefined pattern and often follow the paradigm of end-to-end optimisation.

The seminal work of Xu et al. (2015b) introduced the most popular framework for neural image captioning; that of encoder-decoder with attention over images. Their architecture, which mainly consists of a CNN encoder for the image part and an RNN decoder that generates captions, has inspired a large portion of the subsequent captioning literature. Examples include several types of attention mechanisms, such as top-down and bottom-up attention (Anderson et al., 2018) and adaptive attention (Lu et al., 2017) and also models that incorporate high-level concepts and external information (Wu et al., 2017a) or entities identified by object detectors (Lu et al., 2018). Moreover, the increased performance that image captioning systems achieved has led to the introduction of new tasks (for example, entity-aware captioning Biten et al. 2019), and new, larger and more challenging datasets, such as the Conceptual Captions dataset (Sharma et al., 2018). More recently, given the prevalence of Transformer models (Vaswani et al., 2017) in several NLP tasks, there has been a shift of image captioning systems from the CNN-RNN encoder-decoder framework to Transformer encoder-decoders (Zhu et al., 2018; Yu et al., 2019; Li et al., 2019).

While, usually, captioning methods treat images as sets of objects identified in them (bags of regions), there has also been work that uses structural image cues or relations. An excellent example of such cues are visual dependency representations (Elliott and Keller, 2013), which can be used to outline what can be described as the visual counterpart of dependency trees. Another example, which has lately started to gain popularity as an image representation formalism, is scene graphs (Johnson et al., 2015), that encode objects, attributes and relationships between objects in an image.

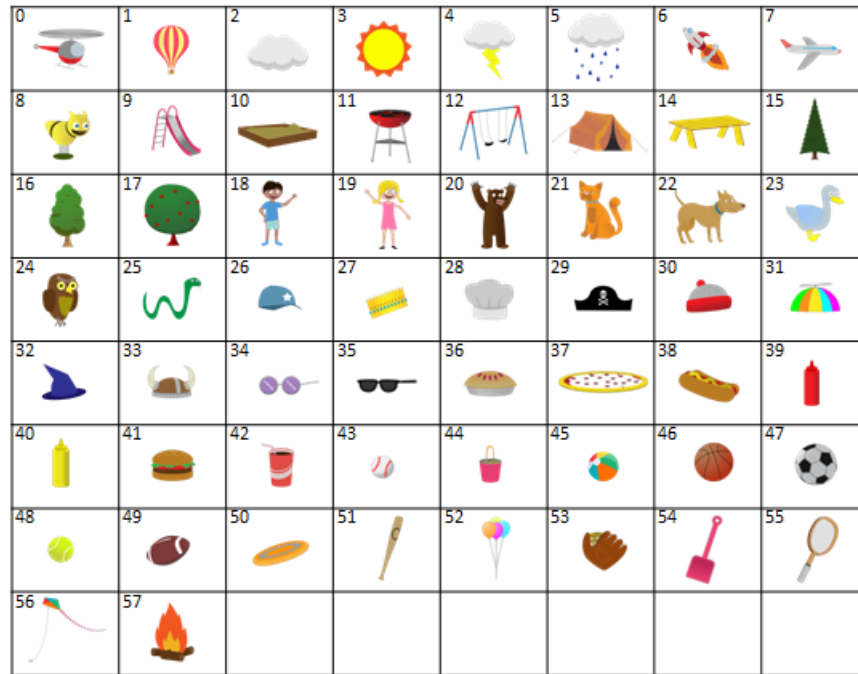


Figure 4.3: The set of objects and cliparts used in the Abstract Scenes Dataset. Some of these objects can have different manifestations (for example, the girl can be sad).

#### 4.1.1 Abstract Scenes

While computer vision advances have given an unprecedented potential to image description generation, performance on image understanding tasks affects the caption generation process, as those two problems are commonly solved together in a pipeline or a joint fashion. To countermeasure that, Zitnick and Parikh (2013) introduced the notion of “abstract scenes”, that is abstract images generated by stitching together clipart images. Their intuition is that working on abstract scenes can allow for a more clean and isolated evaluation of caption generators and also lead to relatively easy construction of datasets of images with semantically similar content.

To this end, they introduced and released the Abstract Scenes Dataset,<sup>1</sup> which contains a total of 10,020 scenes. Each scene is represented as a set of clipart objects (see Figure 4.3 for a full list) placed in different positions and sizes in a background image (grassy area and sky). Cliparts can appear in different ways, for example, the boy and the girl (cliparts 18 and 19) can be depicted as being sad, angry, sitting or running. Finally, each scene has up to eight different descriptions, collected through crowdsourcing. An example image of this dataset along with corresponding descriptions can be seen in Figure 4.4.

<sup>1</sup><https://vision.ece.vt.edu/clipart/>



- |                                      |                                     |
|--------------------------------------|-------------------------------------|
| 1. mike is kicking the soccer ball   | 4. jenny is kicking the soccer ball |
| 2. mike is sitting on the cat        | 5. the sun is behind jenny          |
| 3. jenny is standing next to the dog | 6. the soccer ball is under the sun |

Figure 4.4: An image with six descriptions (captions) from the Abstract Scenes Dataset.

The Abstract Scenes Dataset has been used for description generation (Ortiz et al., 2015), sentence-to-scene generation (Zitnick et al., 2013) and object dynamics prediction (Fouhey and Zitnick, 2014). The importance of this dataset and the work around it is twofold: firstly, it evaluates the effect of perfect image recognition and semantic image information on corresponding tasks. Secondly, it sets an example in favour of dissecting and thoroughly investigating problems and subproblems, contradicting the common practice of unjustifiably applying end-to-end architectures.

### 4.1.2 Evaluation

The evaluation of image description generation is fundamentally challenging and fuzzy (Reiter and Belz, 2009). This inherent difficulty stems from the fact that there is no universal standard on what constitutes a good (or a good enough) caption. Apart from the obvious features of grammatical correctness and text coherence that a caption must possess, the variability in acceptable content leads to a large number of potential captions per image. Captions describing just a part of the image (such as Caption 2 of Figure 4.1, which mentions the boy and the teddy bear, but fails to include the doughnut) or stating a correct but not central to the scene fact (Caption 6 in Figure 4.4 focuses on an aspect of the image that is not particularly interesting) cannot be eagerly dismissed as “wrong”, but are admittedly not as good as more complete ones. This makes any evaluation that compares automatically generated captions against gold captions from a dataset, unidirectional: high accuracy implies good performance, while low accuracy cannot provide any insight about the quality of the

output (it can be either bad, or not similar to the captions in the dataset).

#### 4.1.2.1 Automatic Evaluation

Being essentially a Natural Language Generation (NLG) task, image description generation is automatically evaluated by metrics that rely on the similarity between system-generated and “correct” captions. The similarity is gauged by calculating co-occurrence statistics between the two captions.

The most popular metric for image captioning is BiLingual Evaluation Understudy (BLEU; Papineni et al. 2001) which was initially introduced to evaluate Machine Translation. BLEU is a corpus-wide metric, which relies on the concept of *modified n-gram precision*. Assuming a set  $\mathcal{C}$  of generated outputs (candidates) and a set  $\mathcal{R}$  of correct (reference) outputs, the modified n-gram precision is

$$p_n = \frac{\sum_{G \in \mathcal{C}} \sum_{u_n \in G} C_{\text{clip}}(u_n)}{\sum_{G' \in \mathcal{C}} \sum_{u'_n \in G'} C(u'_n)}, \quad (4.1)$$

where  $u_N$  is a text unit (n-gram),  $C$  denotes the count function that counts common units between the reference and the candidate and  $C_{\text{clip}}$  returns the minimum between the count and the maximum reference count of a unit, thus ensuring that each unit’s count is capped at the its largest count in all references. Depending on the size  $N$  of the largest unit considered in the evaluation (unigrams, bigrams, trigrams etc.), a different BLEU score is returned. The value of  $N$  characterises the metric: BLEU-2 refers to the metric that uses unigrams and bigrams, while BLEU-3 uses unigrams, bigrams and trigrams. Formally, the BLEU- $N$  score is defined as

$$\text{BLEU-N} = \text{BP} \cdot \exp \left( \sum_{n=1}^N w_n \log p_n \right), \quad (4.2)$$

where BP denotes a brevity penalty. The calculation of the brevity penalty involves the length of each candidate output  $|G|$  and the effective reference length  $r$ . The effective reference length is the sum of the best match lengths (the length of the candidate set that is closest to the “correct” length) for each candidate sentence in the corpus. The BP is then calculated as follows:



$$\text{BP} = \begin{cases} 1, & |G| > r \\ e^{1-r/|G|}, & |G| \leq r. \end{cases} \quad (4.3)$$

Apart from BLEU, which is traditionally included in most image description evaluations, other metrics are also used:

- **Metric for Evaluation of Translation with Explicit ORdering (METEOR; Banerjee and Lavie 2005)** was introduced as a language specific metric for Machine Translation. It uses linguistic resources and uses a set of parameters learnt by human judgements in several translation pairs. It is based on precision, recall and F1 scores, calculated over the number of common units between aligned reference and candidate pairs.

Formally, METEOR is defined as

$$\text{METEOR} = F_m \cdot (1 - p), \quad (4.4)$$

where  $F_m = \frac{10PR}{R+9P}$ , with P denoting precision and R recall. Also, p is a length penalty calculated as follows

$$p = 0.5 \left( \frac{c_h}{u_m} \right)^3. \quad (4.5)$$

In the above equation,  $c_h$  refers to the number of common consecutive unigrams, and  $u_m$  to the number of matched unigrams between the reference and the system sentence.

- **ROUGE**, the metric proposed for the evaluation of summarisation systems, is also used as a text generation metric. For a detailed description, see Section 3.1.3.1.
- **Consensus-based Image Description Evaluation (CIDEr; Vedantam et al. 2015)** was specifically developed for the task of image captioning. CIDEr encodes the candidate and reference descriptions using TF-IDF vectors and in turn calculates the average cosine similarity between the vectors of a candidate and each of the references. CIDEr combines scores for n-grams with  $n > 1$ , by averaging scores for each value of n. Formally, for a candidate sentence  $c_i$  and a set of k reference sentences

$S_i = \{s_{ij}\}, j \in \{1, 2, 3, \dots, k\}$ , CIDEr is defined as

$$\text{CIDEr}(c_i, S_i) = \sum_{n=1}^N w_n \text{CIDEr}_n(c_i, S_i), \quad (4.6)$$

where  $w_n$  are weights and  $\text{CIDEr}_n$  is defined as

$$\text{CIDEr}_n(c_i, S_i) = \frac{10}{m} \sum_j e^{\frac{-(l(c_i) - l(s_{ij}))^2}{2\sigma^2}} \frac{\min(g^n(c_i), g^n(s_{ij})) \cdot g^n(s_{ij})}{\|g^n(c_i)\| \|g^n(s_{ij})\|}, \quad (4.7)$$

where  $g^n$  is a function that returns a TF-IDF  $n$ -gram representation for each of the inputs,  $l$  is the length function and  $\sigma = 6$ .

- **Semantic Propositional Image Caption Evaluation (SPICE; Anderson et al. 2016)** mostly evaluates the content of captions, hence it follows a different approach. From each reference and candidate caption, a scene graph is extracted; one that could potentially serve as the scene graph of the described image. The final SPICE score is the F1 score calculated using precision and recall of co-occurring objects and relationships between the graphs. The nature of SPICE makes it more relevant as a criterion for accuracy in caption, and less for syntactic or linguistic quality.

#### 4.1.2.2 Human Evaluation

The problem of crafting a perfect metric for image captioning is an open research question (Kilickaya et al., 2017). The correlation between metric values and human judgments has been an object of controversy (Elliott and Keller, 2014), despite the encouraging data presented by the authors of each of the aforementioned metrics. Moreover, it is known that image captioning metrics have blind spots that are relatively easy to game (Cui et al., 2018b).

For that reason, following a recent trend in many tasks, human evaluation is often employed to measure the quality of the the output of image captioning systems. Human experiments strive to determine the quality of captions based on one or more of the following criteria:

- whether the description is accurate.
- whether the description is grammatically correct.
- whether the description is relevant for the image.
- whether the description is creative.

- whether the description is better than those generated by other systems/sources.

Human experiments are invaluable, since they address the fallibility of automatic metrics and can reveal information not clearly reflected in the metric values. However, they may have shortcomings, as discussed in Section 3.1.3.2.

## 4.2 Multi-view Representation Learning for Image Captioning

The present chapter discusses a multi-view approach for image captioning. Our learning module is based on Canonical Correlation Analysis (see Section 2.2.2.1 for a detailed description of the method). This component operates by projecting the inputs and outputs of the training set to a low-dimensional space. The projection ensures that inputs and outputs corresponding to each other are projected to close points in that low-dimensional space, in which decoding happens. As such, our training algorithm builds on previous work by Udupa and Khapra (2010) and Jagarlamudi and Daumé III (2012) who used CCA for transliteration. Our approach of Canonical Correlation Inference is simple to implement and does not require complex engineering tailored to the task. It mainly needs two feature functions, one for the input values and one for the output values and does not require features combining the two. We also propose a simple decoding algorithm when the output space is text. The method discussed is similar to the one we employ for abstractive summarisation (Section 3.4).

We test our learning algorithm on the domain of language and vision. We use the Abstract Scenes dataset of Zitnick and Parikh (2013), with the goal of mapping images (in the form of clipart abstract scenes) to their corresponding image descriptions. This problem has a strong relationship to recent work in language and vision that has used neural networks or other computer vision techniques to solve a similar problem for real images. Our work is most closely related to the work by Ortiz et al. (2015) who used phrase-based machine translation to translate images to corresponding descriptions.

### 4.2.1 Learning

We assume two structured spaces, an input space  $\mathcal{X}$  and an output space  $\mathcal{Y}$ . As usual in the supervised setting, we are given a set of instances  $(x_1, y_1), \dots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$ , and the goal is to learn a decoder  $\text{dec}: \mathcal{X} \rightarrow \mathcal{Y}$  such that  $\text{dec}(x)$  is the “correct” output as learnt

based on the training examples.

The basic idea in our learning procedure is to learn two projection functions  $u: \mathcal{X} \rightarrow \mathbb{R}^m$  and  $v: \mathcal{Y} \rightarrow \mathbb{R}^m$  for some low-dimensional  $m$  (relatively to  $d$  and  $d'$ ). In addition, we assume the existence of a similarity measure  $\rho: \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$  such that for any  $x$  and  $y$ , the better  $y$  “matches” the  $x$  according to the training data, the larger  $\rho(u(x), v(y))$  is. The decoder  $\text{dec}(x)$  is then defined as:

$$\text{dec}(x) = \arg \max_{y \in \mathcal{Y}} \rho(u(x), v(y)). \quad (4.8)$$

Our key observation is that one can use Canonical Correlation Analysis to learn the two projections  $u$  and  $v$ . The learning algorithm assumes the existence of two feature functions  $\phi: \mathcal{X} \rightarrow \mathbb{R}^{d \times 1}$  and  $\psi: \mathcal{Y} \rightarrow \mathbb{R}^{d' \times 1}$ , where  $d$  and  $d'$  could potentially be large, and the feature functions could potentially lead to sparse vectors.

We then apply a modified version of Canonical Correlation Analysis on the two “views:” one view corresponds to the input feature function and the other view corresponds to the output feature function. This means we calculate the following three matrices  $D_1 \in \mathbb{R}^{d \times d}$ ,  $D_2 \in \mathbb{R}^{d' \times d'}$  and  $\Omega \in \mathbb{R}^{d \times d'}$ :

$$\begin{aligned} D_1 &= \text{diag} \left( \frac{1}{n} \sum_{i=1}^n \phi(x_i)(\phi(x_i))^\top \right) \\ D_2 &= \text{diag} \left( \frac{1}{n} \sum_{i=1}^n \psi(y_i)(\psi(y_i))^\top \right) \\ \Omega &= \frac{1}{n} \sum_{i=1}^n \phi(x_i)(\psi(y_i))^\top \end{aligned} \quad (4.9)$$

where  $\text{diag}(A)$  for a square matrix  $A$  is a diagonal matrix with the diagonal copied from  $A$ . We then apply thin singular value decomposition on  $D_1^{-1/2} \Omega D_2^{-1/2}$  so that

$$D_1^{-1/2} \Omega D_2^{-1/2} \approx U \Sigma V^\top, \quad (4.10)$$

with  $U \in \mathbb{R}^{d \times m}$ ,  $\Sigma \in \mathbb{R}^{m \times m}$  is a diagonal matrix of singular values and  $V \in \mathbb{R}^{d' \times m}$ . The

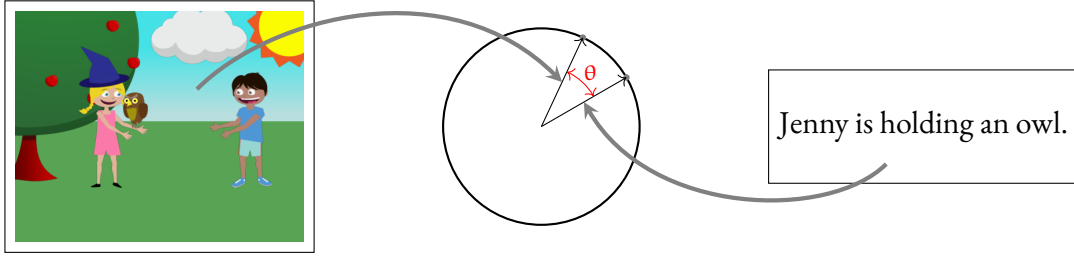


Figure 4.5: Demonstration of CCA inference. An object from the input space  $\mathcal{X}$  (the image on the left  $\mathbf{x}$ ) is mapped to a unit vector. Then, we find the closest unit vector which has an embodiment in the output space,  $\mathcal{Y}$ . That embodiment is the text on the right,  $\mathbf{y}$ . It also holds that  $\rho(\mathbf{u}(\mathbf{x}), \mathbf{v}(\mathbf{y})) = \cos \theta$ .

value of  $m$  should be relatively small compared to  $d$  and  $d'$ . We then choose  $\mathbf{u}$  and  $\mathbf{v}$  to be:

$$\begin{aligned} \mathbf{u}(\mathbf{x}) &= (\mathbf{D}_1^{-\frac{1}{2}} \mathbf{U})^\top \phi(\mathbf{x}), \\ \mathbf{v}(\mathbf{y}) &= (\mathbf{D}_2^{-\frac{1}{2}} \mathbf{V})^\top \psi(\mathbf{y}), \end{aligned} \quad (4.11)$$

which we use as projection functions.

Osborne et al. (2016) showed that CCA maximises the following objective:

$$\sum_{i,j} d_{ij} - n \sum_{i=1}^n d_{ii}^2, \quad (4.12)$$

where

$$d_{ij} = \sqrt{\frac{1}{2} \left( \sum_{k=1}^m (\mathbf{u}(\mathbf{x}_i) - \mathbf{v}(\mathbf{y}_j))^2 \right)}. \quad (4.13)$$

This objective is maximised with respect to the projections that CCA finds,  $\mathbf{u}$  and  $\mathbf{v}$ . This means that CCA finds projections such that the Euclidean distance between  $\mathbf{u}(\mathbf{x})$  and  $\mathbf{v}(\mathbf{y})$  for matching  $\mathbf{x}$  and  $\mathbf{y}$  is minimised, while it is maximised for  $\mathbf{x}$  and  $\mathbf{y}$  that have a mismatch between them.

As such, it is well-motivated to use a similarity metric  $\rho(\mathbf{u}(\mathbf{x}), \mathbf{v}(\mathbf{y}))$  which is inversely monotone with respect to the Euclidean distance between  $\mathbf{u}(\mathbf{x})$  and  $\mathbf{v}(\mathbf{y})$ . We choose cosine similarity as a similarity metric:

$$\rho(z, z') = \frac{\sum_{i=1}^m z_i z'_i}{\sqrt{\sum_{i=1}^m z_i^2} \sqrt{\sum_{i=1}^m (z'_i)^2}} \quad (4.14)$$

$$= \frac{\langle z, z' \rangle}{\|z\| \cdot \|z'\|}. \quad (4.15)$$

It is worth noting that for any two vectors  $z$  (denoting  $u(x)$ ) and  $z'$  (denoting  $v(y)$ ), by simple algebraic manipulation, it holds that

$$-\langle z, z' \rangle = \frac{1}{2} \left( \|z - z'\|^2 - \|z\|^2 - \|z'\|^2 \right). \quad (4.16)$$

Consequently, if the norms of  $z$  and  $z'$  are constant, maximizing the cosine similarity between  $z$  and  $z'$  is effectively the same as minimising the Euclidean distance between  $z$  and  $z'$ . In our case, the norms of  $u(x)$  and  $v(y)$  are not constant, but we find that our algorithm is much more stable when the cosine similarity instead of Euclidean distance is used.

Figure 4.5 shows an outline of our CCA inference algorithm and Algorithm 4.1 presents it in detail.

### 4.2.2 Decoding

While the described approach to map from an input space to an output space through CCA is rather abstract and general, decoding is not always trivial. This is regardless of  $\mathcal{X}$  – once  $x$  is given, it is mapped using  $u(x)$  to a vector in  $\mathbb{R}^m$ , and at this point this is the information we use to further decode into  $y$  – the structure of  $\mathcal{X}$  before this transformation does not change much the complexity of the problem.

In order to generate image captions we use a decoder similar to the one presented in Section 3.4.1 for abstractive summarisation. Concretely, this decoder is a Metropolis Hasting algorithm that iteratively refines a caption, while ensuring that generated captions have high cosine similarity with the generated representation of the image. For reference, we present the algorithm of the decoder in Listing 4.2.

In a set of preliminary experiments, we found that while our algorithm gives adequate descriptions to the images, it is not unusual for it to give short descriptions that just mention a single object in the image. This relates to the adequacy-fluency tension that exists in machine translation problems. To overcome this issue, we add to the cosine similarity a term

---

**Algorithm 4.1** The CCA learning algorithm.
 

---

**Input:**
 $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$  ▷ set of  $N$  examples:  $i \in \{1, \dots, N\}$ 
 $m$  ▷ an integer
 $\phi(x)$  and  $\psi(y)$  ▷ two feature functions
1: **procedure** CCA\_LEARNING2:   calculate  $\Omega \in \mathbb{R}^{d \times d'}$ 

$$\Omega_{ij} = \sum_{k=1}^n [\phi(x_k)]_i [\psi(y_k)]_j$$

3:   calculate  $D_1 \in \mathbb{R}^{d \times d}$  such that  $(D_1)_{ij} = 0$  for  $i \neq j$  and

$$(D_1)_{ii} = \sum_{k=1}^n [\phi(x_k)]_i [\phi(x_k)]_i$$

4:   calculate  $D_2 \in \mathbb{R}^{d' \times d'}$  such that  $(D_2)_{ij} = 0$  for  $i \neq j$  and

$$(D_2)_{ii} = \sum_{k=1}^n [\psi(y_k)]_i [\psi(y_k)]_i$$

5:   calculate  $m$ -rank thin SVD on  $D_1^{-\frac{1}{2}} \Omega D_2^{-\frac{1}{2}}$ :  $D_1^{-\frac{1}{2}} \Omega D_2^{-\frac{1}{2}} \approx U \Sigma V^\top$ 6:    $u(x) = (D_1^{-\frac{1}{2}} U)^\top \phi(x)$ 7:    $v(y) = (D_2^{-\frac{1}{2}} V)^\top \psi(y)$ 8:   **return**  $u(x), v(y)$ 


---

$\eta|y|$  where  $\eta$  is some positive constant tuned on a development set and  $|y|$  is the length of the sampled sentence. This pushes the decoding algorithm to prefer textual descriptions which are longer, effectively avoiding the brevity penalty of automatic metrics, while making sure that the output is not overly trivial (e.g. one-word captions). This scoring method is slightly different than the one used for abstractive summarisation and which is described in Equation 3.17.

## 4.3 Experiments

This section presents our experiments on generating descriptions for abstract scenes using our proposed method.

---

**Algorithm 4.2** The CCA decoding algorithm.
 

---

**Input:**

$x$	▷ an input example
$\rho$	▷ a similarity metric
$u, v$	▷ two projection functions
$Q$	▷ a probabilistic phrase table
$\eta \geq 0$	▷ a constant
$\tau \in (0, 1)$	▷ a constant
$T$	▷ a starting temperature

1: **procedure** CCA\_DECODER2:   Let  $y^*$  be an arbitrary point in the output space.3:    $y' \leftarrow y^*$ .4:    $t \leftarrow T$ .5:   **while** temperature  $t$  is below a given value **do**6:     uniformly choose two different integers  $i$  and  $j$ , between 1 and  $|y'|$ 7:     choose randomly a phrase  $p$  from  $Q(p \mid y'_{i-1}, y'_{j+1})$ 8:      $y \leftarrow y'_1 \cdots y'_{i-1} p y'_{j+1} \cdots y'_{|y'|}$ 9:     **if**  $\rho(u(x), v(y)) + \eta|y| \geq \rho(u(x), v(y^*)) + \eta|y^*|$  **then**10:        $y^* \leftarrow y$ 11:        $\alpha_0 \leftarrow \frac{\exp\left(\frac{1}{t}\rho(u(x), v(y)) + \eta|y|\right)}{\exp\left(\frac{1}{t}\rho(u(x), v(y')) + \eta|y'|\right)}$ 12:        $\alpha_1 \leftarrow \frac{|y|^2 Q(y'_1 \cdots y'_j \mid y'_{i-1}, y'_{j+1})}{|y'|^2 Q(y_i \cdots y_j \mid y_{i-1}, y_{j+1})}$ 13:        $\alpha \leftarrow \{1, \alpha_0 \cdot \alpha_1\}$ 14:       uniformly sample a number  $k$  from  $[0, 1]$ 15:       **if**  $k < \alpha$  **then**16:          $y' \leftarrow y$ 17:        $t \leftarrow \tau t$ **return**  $y^*$ **4.3.1 Experimental Setup**

We use the same data split as Ortiz et al. (2015), with 7,014 of the scenes as a training set, 1,002 as a development set and 2,004 as a test set.<sup>2</sup> Each scene is labeled with at most eight

---

<sup>2</sup>Our dataset splits and other information can be found in <http://cohort.inf.ed.ac.uk/canonical-correlation-inference.html>.



y	p	y'	probability
waiting	to	get	1.000
with	the	bucket.	0.750
pizza	on	the	0.343
trying	to get away from	jenny	0.050
baseball	with	the	0.033
is	playing near the	swings.	0.011
< begin >	jenny is playing with a	colorful	0.008
is	surprised by the	owl	0.006
mike	and the bear are	standing	0.002

Table 4.1: Example of phrases and their learnt probabilities for the function  $Q(p | y, y')$ . The marker < begin > marks the beginning of a sentence.

short captions. We use all of these captions in the training set, leading to a total of 42,276 training instances.

**Input Spaces** The feature function  $\phi(x)$  for an image is based on the “visual features” that come with the abstract scene dataset. More specifically, these are binary features that fire for 11 object categories, 58 specific objects, co-occurrence of object category pairs, co-occurrence of object instance pairs, absolute location of object categories and instances, absolute depth, relative location of objects, relative location with directionality the object is facing, a feature indicating whether an object is near a child’s hand or a child’s head and attributes of the children (pose and facial expression). The total number of features for this function is 7,149.

We define the feature function  $\psi(y)$  for image description as one returning one-hot representations for all phrases from the phrase table  $\mathcal{P}$  that fire in the image. We use the phrase table of [Ortiz et al. \(2015\)](#), which was obtained through the Moses toolkit ([Koehn et al., 2007](#)) and contains 30,911 phrases. The size of the domain of  $Q$  (the size of the phrase table with context words) is 120,019. Table 4.1 gives a few example phrases and their corresponding probabilities.

**Output Dimensionality** For the CCA learning algorithm, we also need to decide on the output dimensionality. We varied values between 30 and 300 (in steps of 10) and tuned on

the development set by maximizing BLEU score against the set of references.<sup>3</sup> Interestingly, the BLEU scores did not change much (they usually were within one point of each other for sufficiently large values), pointing to a stability of the algorithm with respect to the number of dimensions used.

### 4.3.2 Results

We compare the performance of our model with that of several baselines and systems from the work of [Ortiz et al. \(2015\)](#), reporting BLEU and METEOR scores. More specifically, we consult the following systems:

- **LBL** is a log-bilinear language model trained on the image captions only.
- **MLBL** is multimodal log-bilinear model, implementation of [Kiros et al. \(2014\)](#).
- **Image Retrieval** refers to a system that for every test image, queries the set of training images for the most similar one, and returns a random description of that training example.
- **CCA Retrieval**. We report scores of a retrieval baseline based on representations learnt by our learning module. Similarly to *Image Retrieval*, for each test image, we search for the most similar caption from the training set, with similarity calculated on the shared learnt space.
- **Keyword** is a system that annotates every image with keywords that most probably describe it and then do a search query against all training data descriptions, returning the description that is closest (in terms of TF-IDF similarity) to the keywords. Keyword assignment is accomplished using logistic regression and features based on clipart objects and text information (POS tags, objects, dependency roles).
- **Template** uses templates inferred from dependency parses of the training data descriptions. A set of templates is discovered and a classifier that associates images with templates is trained. At test time, a template is assigned to every image and the slots of the templates are filled by the model for keyword assignment described in the Keyword system.
- **SMT** is the Statistical Machine Translation (SMT) system of [Ortiz et al. \(2015\)](#). It first selects pairs of clipart objects that are important enough to be described by solving an integer linear programming problem and creates a “visual encoding” using a visual

---

<sup>3</sup>We use the `multeval` package from <https://github.com/jhclark/multeval>.

dependency grammar (Elliott and Keller, 2013). Finally, it translates the latter to a sentence, using a phrase-based SMT engine.

The scores of all systems are given in Table 4.2. Note that we also experimented with a sequence-to-sequence neural model,<sup>4</sup> which turned out to perform poorly, giving a BLEU score of 10.20 and a METEOR score of 15.20 and largely inappropriate captions. We conject that sequence-to-sequence models are unfit for this dataset, probably due to its size; Rastogi et al. (2016) also report similar results. While our CCA system scores better than most of the baselines, it does not achieve as high scores as the machine translation system.

While interpreting the results of the experiments, it is important to keep in mind that the captions in the dataset, as well as those generated by the different systems, are not complete. Each of them describes a specific aspect of each scene, as discussed previously and shown in the captions of Figure 4.8. As such, the use of machine translation metrics such as BLEU and METEOR, or any word overlap metric for that matter, is not necessarily the best way to identify the correctness of a textual description. To demonstrate this point, we measure BLEU scores of one of the reference sentences while comparing it to the other references in the set. We did that for each of the eight batches of references available in the training set. The average reference BLEU score is 24.1 and the average METEOR score is 20.0, a significant drop compared to the SMT system. We conclude that the SMT system is not “creatively” mapping images to corresponding descriptions. Instead, it relies heavily on the training set captions, and learns how to map images to sentences in a manner which does not generalise very well outside of the training set.

An indication that our system creates a more diverse set of captions is that the number of *unique* captions it generates for the test set is significantly larger than that of the SMT system. The latter generates 359 unique captions (out of 2,004 instances in the test set), while our CCA system generates 496 captions, an increase of 38.1%.

### 4.3.3 Human Evaluation

To test our hypothesis about caption diversity and quality, we conducted the following human experiment. We asked 12 subjects to rate the captions of 300 abstract scenes.<sup>5</sup> Each rater was presented with three captions: a reference caption (selected randomly from the

---

<sup>4</sup>We used a modified version of the sequence-to-sequence Tensorflow model: <https://github.com/tensorflow/nmt>.

<sup>5</sup>The ratings can be found on <http://cohort.inf.ed.ac.uk/canonical-correlation-inference.html>.

	system	BLEU	METEOR
Ortiz et al.	LBL	7.3	17.7
	MLBL	12.3	20.4
	Image	12.8	21.7
	CCA retrieval	13.0	20.1
	Keyword	14.7	26.6
	Template	40.3	30.4
	SMT	43.7	35.6
	CCA	26.1	25.6

Table 4.2: Scene description evaluation results on the test set, comparing the systems from Ortiz et al. to our CCA inference algorithm (the first six results are reported from the Ortiz et al. paper). The CCA result uses  $m = 120$  and  $\eta = 0.05$ , selected after tuning on the development set.

gold-standard captions), an SMT caption and a caption from our system (presented in a random order) and was asked to rate the captions on adequacy (using a scale of 1 to 5).

The instructions given to raters are shown below. The rankers were introduced to the task by a webpage stating the following text and presenting the examples shown in Figure 4.6.

*Welcome!*

*Once you read the following instructions, click [here] to start ranking scenes!*

*In this task you will look at a series of images and image descriptions (captions) created by a computer program. At every step, you will be presented with images and three (3) descriptions for each image. You will be asked to judge whether the descriptions are relevant (accurate and informative) to the image.*

*You will rate every description in 1-5 rating scale, where 5 is the highest possible rating and 1 is the lowest. Please keep in mind that:*

- ***There are no correct answers***, feel free to choose the rating that feels most appropriate - it is a valid response.
- ***The objective of the rating is not to mark the best possible description. You can give high score to more than one description if they are accurate and informative.***

*Thank you for your participation.*

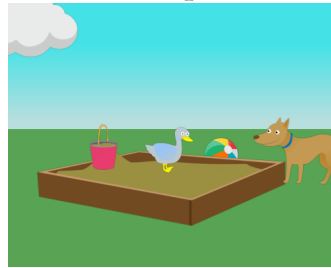
Most images were rated exactly twice, with a few images getting three raters. A score of 1 or 2 means that the caption likely does not adequately describe the scene. A score of 3 usually

### Examples



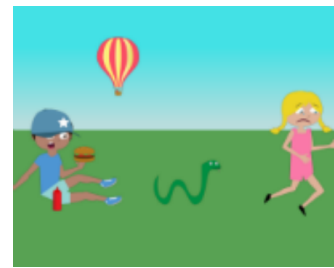
*Mike and Jenny are playing ball.*

This description will get a high rating (4 or 5), since it is accurate and captures a significant aspect of the image.



*Mike is in the sandbox.*

This isn't a good description, as it only marginally relates with the image.



*Mike is holding a hamburger.*

While the description is accurate and relevant to the image, it fails to capture the most interesting or central aspect of the image. Therefore, it should be given a medium rating (around 3).

Figure 4.6: Examples shown to annotators for image captioning human evaluation.

means that the caption describes some salient component in the scene, but perhaps not the most important one. Scores of 4 and 5 usually denote good captions that adequately describe the corresponding scenes. This experiment is similar to the one done by [Ortiz et al. \(2015\)](#). The ranking results are given in [Table 4.3](#). The results show that our system tends to score higher for images which are highly ranked (by both the SMT system and CCA), but tends to score lower for images which are lower ranked.

In addition, we checked the scores for highly ranked captions both for SMT and CCA (ranking larger than 4). For SMT, the BLEU scores are 49.70 (METEOR 40.10) and for CCA it is 41.80 (METEOR 33.10). This is not the result of images in SMT being ranked higher, as the average ranking among these images is 4.18 for the SMT system and 4.25 for CCA. The lower score for CCA indicates that our system generates captions which are not necessarily aligned with the references, but correct nevertheless.

This observation also highlights and perpetuates the controversy around using machine translation evaluation metrics for this dataset and problem. To further support this point, in [Figure 4.7](#) a scatterplot of BLEU versus human rating is provided. It can be seen that the

slice	$r_c < 3$	$r_c \geq 3$
$r_s < 3$	S: 1.77	S: 1.92
	C: 1.64	C: 3.71
$r_s \geq 3$	S: 3.42	S: 3.46
	C: 1.47	C: 3.54

Table 4.3: Average ranking by human judges for cases in which the caption has an average rank of 3 or higher and when its average rank is lower than 3, for both CCA and SMT. Here,  $r_s$  stands for SMT rating,  $r_c$  for CCA rating, “S” for SMT average rank and “C” for CCA average rank. The shaded areas (top left and bottom right) show scores for examples where both systems score low and high respectively.

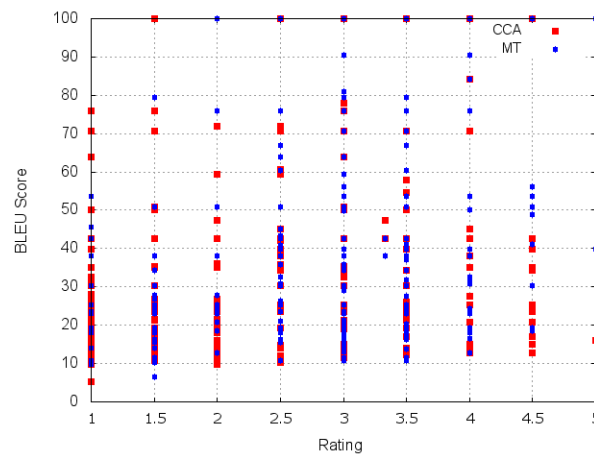


Figure 4.7: Scatter plot of BLEU scores versus human ratings, for both CCA and SMT methods.

correlation between BLEU scores and human ranking is not high; specifically, the correlation between the x-axis (ranking) and y-axis (BLEU scores) for CCA is 0.3 and for the SMT system 0.31. Following work on evaluation for image captioning (Hodosh et al., 2013; Elliott and Keller, 2014; Vedantam et al., 2015), we use sentence-level BLEU scores, although BLEU is generally regarded as a corpus-level metric, especially when used to evaluate machine translation.

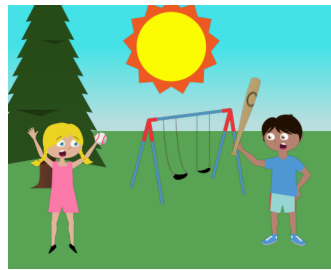
Figure 4.8 presents six examples of generated outputs. The first row includes examples for which the human judges rated the SMT system highly and the CCA system poorly, while the second row presents the reverse case (CCA output ranked high, SMT ranked low).

## 4.4 Conclusions

We describe a technique to predict structures from complex input spaces to complex output spaces based on Canonical Correlation Analysis. Our approach projects the input space into a low-dimensional representation, and then converts it back into an instance in the output space. We demonstrate the use of our method on the structured prediction problem of attaching textual captions to abstract scenes. Human evaluation of these captions demonstrates that our approach is promising for generating text from images.



S: jenny is waving at mike  
C: mike and jenny are camping



S: jenny is wearing a baseball  
C: mike is holding a bat



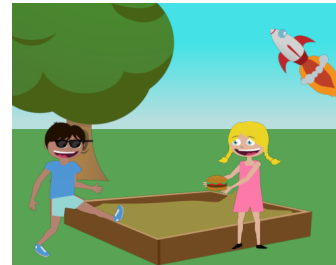
S: jenny is holding a frisbee  
C: jenny is throwing the frisbee



S: jenny is kicking the soccer ball  
C: mike is kicking a ball



S: jenny is holding a hot dog  
C: jenny wants the bear



S: jenny is holding a hamburger  
C: the rocket is behind mike

Figure 4.8: Examples of outputs from the SMT system (S) and CCA inference (C). The top three images give examples where the CCA inference outputs were rated highly by human evaluators (4 or 5), and the machine translation ones were rated poorly (1 or 2). The bottom three pictures show the reverse case.

## Chapter 5

# Multi-view Inference for Movie Understanding

*The world is generating and consuming an enormous amount of video content. Currently on YouTube, people watch over 1 billion hours of video every single day.*

*(The 2nd YouTube-8M Video Understanding Challenge Kaggle Description)<sup>1</sup>*

The consumption of videos has become a common part of the everyday activities of millions of users, either as a form of communication (via mobile devices and social media), a marketing tool (advertisements) or an entertainment tool (streaming services). It is estimated that by 2022, online videos will make up more than 82% of all consumer internet traffic (Cisco, 2019). The unprecedented ease with which video content is generated, posted and shared has brought about the need for efficient and intelligent multimedia processing. Information overload (the state of facing such an amount of information that makes organisation or reasoning difficult; see Chapter 3 for details) and the methods and algorithms devised to tackle it apply equally to textual and multimedia data.

Videos in particular pose a highly complex representational problem for machine learning algorithms. This is not only due to their multimodal nature (combination of image and audio), but also due to the large amount of information they carry. Videos fundamentally consist of several images (frames) played one after another at a high rate. A commonly accepted frame rate and one used in the majority of cinemas is 24 frames per second (fps); high definition equipment has made possible the recording and playback of videos in rates as high as 120 fps, though. Consequently, the amount of information in video files makes

---

<sup>1</sup><https://www.kaggle.com/c/youtube8m-2018>, accessed June 10, 2019.



their processing exceptionally data hungry.

Moreover, the temporal nature of videos introduces an extra level of complexity for representation learning. Based on the fact that video is composed of consecutive frames, video processing essentially relies on the extraction of snapshots and the combination of information from them. Several representational issues arise from this type of modelling: what is the best way to select frames to operate on? Which frames are more important? What kind of information should be extracted from each of the frames? How is information from different modalities combined to create a coherent representation? How is information from different timestamps combined for inference?

Most of those questions do not have trivial answers. Furthermore, there are substantial differences in purpose and content between communication and entertainment videos, that call for particular treatment of each of them. Movies and series<sup>2</sup> normally follow an elaborate plot, making automatic understanding quite challenging. In this chapter, we present a set of tasks that can be thought of as belonging to the class of *shallow movie understanding*; that is tagging tasks that require inference over the episodes of a series. Specifically, we use the CSI dataset (Fermann et al., 2018), which consists of episodes of the Crime Series Investigation (CSI) television series and allows for the modelling of three tagging tasks on three different levels: structural level (enumerating and detecting crime cases), dialogue level (figuring out speaker information) and plot level (identifying perpetrators).

Most existing multi-view representation learning approaches are tested in an unsupervised setup where the representations are learnt separately from the task, and are designed to accommodate the learning of representations for monolithic (albeit multi-view) data points, not sequences (see Wang et al. 2015a for a survey). In this chapter, we propose a neural architecture coupled with a novel training objective that integrates multi-view information for sequence prediction problems. Our model creates a multimodal embedding for every element of a sequence and makes token-level predictions based on those embeddings. Our training objective combines a supervision-guided term (cross-entropy) with a multi-view correlation objective on the available modalities.

Our work extends the set of multi-view counterparts of popular sequence models, many of which have been already mentioned in Section 2.3. Our proposed model enforces correlation between the representations of the available modalities; a technique that has been studied also for non-sequential neural models (Wang et al. 2015a; Chang et al. 2018; also Section 2.2.1).

---

<sup>2</sup>We use the term “series” to refer to episodic shows, broadcast via television or other channels.

Our model bears similarities to the work of [Rajagopalan et al. \(2016\)](#), who propose a general architecture that provides a degree of flexibility in designing different multi-view LSTM cells according to the application at hand, the work of [Ren et al. \(2016\)](#), who propose a multi-modal variant of LSTM and apply it to the task of speaker identification and that of [Zadeh et al. \(2018a\)](#), who use an attention module and a multi-view gated memory to capture and summarise inter-modality interactions (see Section 2.3.2 for a detailed account of sequential multi-view models).

The contributions of this chapter are the following:

- We propose a multi-view sequential inference neural architecture and use a novel training objective for training an RNN consisting of correlational GRU cells.
- We highlight the importance of multi-view fusion for multimodal applications, by comparing our model with a non-multi-view variant that employs multi-head supervised attention to make use of both the sentence-level and the token-level perpetrator annotations of the dataset.
- We introduce two novel tasks pertaining to shallow movie understanding that can be tackled in the context of television series data. We empirically show the effectiveness of our architecture and training objective on the perpetrator mention identification and the two newly introduced tasks, by using the Crime Scene Investigation (CSI) television series dataset. Notably, for the perpetrator identification task, our model significantly outperforms previous work ([Fermann et al., 2018](#)).

## 5.1 Movie Understanding

Video understanding is primarily a domain of interest of Computer Vision. Examples of tasks that can be applied to videos include automatically identifying and/or following objects ([Wu et al., 2015](#); [Caelles et al., 2019](#)), people ([Zheng et al., 2016](#)) or identifying actions ([Wang et al., 2016](#)), events ([Monfort et al., 2019](#)) or emotion ([Zadeh et al., 2018b](#)). The presence of speech in many videos though makes video datasets attractive to the NLP community, too. Drawing parallels from speech processing, and incorporating image and audio features, NLP methods can be applied to videos that contain a substantial amount of speech.

Conversely, even though many NLP problems concern exclusively textual or speech data, it has been shown that integrating multimodal information (such as images, video or audio) is beneficial for a variety of tasks. For example, visual information has been used in affect

analysis (Kahou et al., 2016), sentiment analysis (Morency et al., 2011) and machine translation (Calixto et al., 2017; Lala and Specia, 2018). This is also the case for problems which are sequential in nature, such as video summarisation (Smith and Kanade, 1998), continuous prediction of affect (Nicolaou et al., 2011) or engagement level prediction (Rehg et al., 2013).

Lately, work on video understanding has expanded to cover not only simple, short-length videos, but also long videos with an, often, complex plot, created for entertainment purposes. Given their popularity and consequent expansion in number, series video data can benefit multimodal machine learning research and applications. Series commonly span many episodes (organised in loosely or tightly connected seasons), providing a large amount of data that data-hungry models can take advantage of. The sheer volume of data gives rise to several practical problems that machine learning models can tackle, such as the segmentation of continuous video streams to semantically coherent fragments (Del Fabro and Böszörményi, 2013) and speaker diarisation (Miró et al., 2012; Bredin et al., 2014).

Work on movie and series analysis can be classified into three broad categories:

- *Deep semantic understanding* includes tasks that require thorough content analysis and reasoning, for example movie question answering (Tapaswi et al., 2016; Kim et al., 2017), movie description (Rohrbach et al., 2016) or overview generation (Gorinski and Lapata, 2018).
- *External understanding* refers to tasks whose end goal is not the analysis of the video content itself, but meta-information extraction relevant to preferences and recommendations for consumers of the videos (Bennett et al., 2007; Shi et al., 2013; Yang et al., 2012).
- *Shallow understanding* refers to tasks that operate on the content level and extract content-related information, albeit without requiring complex reasoning. Their output is more tailored to structured prediction. Example tasks include speaker identification (Knyazeva et al., 2015), movie segmentation (Liu et al., 2013) and perpetrator mention identification (Frermann et al., 2018). While these tasks are not trivial, their tagging nature makes them less demanding than those listed under deep understanding.

We choose to tackle three problems of shallow understanding, namely a crime case sequence tagging problem, a speaker type sequence tagging problem and the problem of perpetrator mention identification. Inference on multimodal sequences can take the form of inferring a label for a whole sequence, or a label for each of the parts of it; since all three problems

refer to tagging sequence elements, we employ a multi-view, multimodal, sequential inference framework for them. We demonstrate the effectiveness of our model in an incremental inference setup, wherein it makes predictions on the fly, without encoding the sequence in full, following a realistic scenario of interacting with data. This is a critical feature for online applications such as simultaneous translation (interpretation) and also a desirable behaviour for movie processing models that mimic a human viewer watching a movie for the first time.

## 5.2 Multi-view Sequential Inference

Taking together the idea of multi-view learning with incremental sequence labeling, we formulate our problem as follows.

We assume a set  $\mathcal{X}$  of  $M$  examples. Every  $X_j \in \mathcal{X}$ ,  $j \in \{1, \dots, M\}$ , consists of  $T$  elements, which form a sequence  $X_j = [x_{j1} \ x_{j2} \ \dots \ x_{jT}]$ . Sequences with less than  $T$  elements are padded to be of length  $T$ . Each of the elements in the sequence is paired with a label from a set  $\mathcal{Y}$  (binary or multi-class), which is the desired output. Lastly, for every  $x_{jt}$ , information from a set  $\mathcal{V}$  of different views is available.

More specifically, we consider a dataset where each  $X_j$  is a video and we model three different views/modalities: image, audio and text (from aligned script or subtitles, if available). Each video  $X_j$  is represented as a sequence of short, semantically coherent snippets  $x_{jt}$  (for instance, snippets may correspond to subtitle sentences). For each sequence element and for each of the views  $\mathcal{V} = \{\text{image, audio, text}\}$  we have a feature vector  $x_{jt}^{(k)}$ , with  $k \in \{1, 2, 3\}$  indexing the different views. In addition, we have labels  $y_{jt}$ . The problem is to infer the correct label  $y_{jt}$  for each  $x_{jt}$  at the time it presents itself: data points appear sequentially in a time series and the label prediction should be done using information from the past and the present elements only.

We use a sequence model for incremental modeling of sequential multi-view data. The overall architecture of such a system is that of a RNN. A general time-unfolded overview of the architecture for a single example  $x \in \mathcal{X}$  is shown in Figure 5.1. Input segments  $x_t^{(k)}$  of different views are fed to a cell at every time step, and a single vector  $h_t$  combining information from all views is generated. This embedding is fed to an output layer, which in turn outputs a prediction.

One of the most important and distinguishing features of multi-view models is the framework it uses for the fusion of the different views. We use a multimodal GRU cell, the corre-

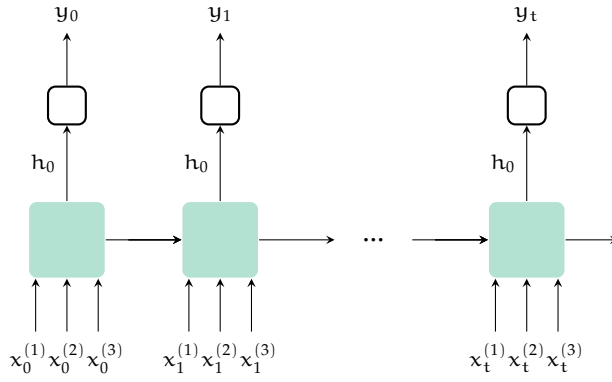


Figure 5.1: General unfolded overview of a multi-view sequence model for an example  $x \in \mathcal{X}$  with three views. At each time step  $t \in \{0, 1, \dots, T\}$ , representations  $x_t^{(k)}$  for each  $k \in \{1, 2, 3\}$  views are fed to a cell. The cell outputs a joint representation  $h_t$  for all the views which is fed to a softmax layer that generates a prediction  $y_t$ .

lational GRU (corrGRU; Yang et al. 2017) as a base for experimentation. The cell takes embeddings  $x_t^{(k)}$  for each view  $k$  and time step  $t$  and after passing each view from a designated GRU cell, it calculates a multi-view embedding for the time step by passing information of all views through the gates of a separate GRU. The output of the view-specific GRUs is used to calculate the Pearson correlation between the different views; the output of this calculation is added as a separate term to be maximised in the total loss. In conclusion, view fusion is achieved not only by the design of the multimodal GRU cell itself (weighted sum of view representations and one common hidden representation), but also by the maximisation of correlation between the views. Section 2.3.4 contains a detailed description of the corrGRU, while Figure 2.13 presents the architecture of the corrGRU cell.

Since correlation can be calculated only for a pair of variables, we generalise the model to more than two views by calculating the total correlation loss as the sum of the correlation losses between all pairs of elements of  $\mathcal{V}$ . Formally, in each step  $t$ , the correlation between the views is calculated as

$$c_t = \sum_{\substack{k, \ell \in \mathcal{V} \\ k \neq \ell}} \frac{\sum_{i=1}^L (h_{it}^{(k)} - \bar{H}_t^{(k)})(h_{it}^{(\ell)} - \bar{H}_t^{(\ell)})}{\sqrt{\sum_{i=1}^L (h_{it}^{(k)} - \bar{H}_t^{(k)}) \sum_{i=0}^L (h_{it}^{(\ell)} - \bar{H}_t^{(\ell)})}}, \quad (5.1)$$

where  $i$  spans over the  $L$  elements of each mini-batch,  $h_{it}^{(k)}$  is the hidden state calculated by the view-specific GRU for the example  $i$ , view  $k$ , at timestep  $t$ . Moreover,  $\bar{H}_t^{(k)} = \frac{1}{L} \sum_{i=1}^L h_{it}^{(k)}$ .

The correlation term is then defined as

$$\mathcal{L}_{\text{corr}} = -\frac{1}{|\mathbb{T}|} \sum_{t \in \mathbb{T}} c_t, \quad (5.2)$$

that is the average of the loss calculated for every time step  $t$ . In order to maximise correlation, the negative sum is used.

Using the correlation loss term, we train our network by jointly minimizing the following objective

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda \mathcal{L}_{\text{corr}}, \quad (5.3)$$

where  $\mathcal{L}_{\text{CE}}$  stands for cross-entropy loss,  $\mathcal{L}_{\text{corr}}$  is the correlation loss term, and  $\lambda$  weights the contribution of  $\mathcal{L}_{\text{corr}}$  to the total loss. This compound objective function is one of the distinctive features of our approach. It enables the model to take advantage of the labels available from the dataset, and at the same time optimise for the correlation between the available views. The underlying idea is that the constraint of the correlation will push the model to create more informative embeddings than those it would create if the cross-entropy loss was used alone.

**Multi-head Attention** Attention mechanisms (Bahdanau et al., 2015), in various forms, have been used in several multimodal applications, such as sentiment analysis, speaker trait recognition and emotion recognition (Zadeh et al., 2018c), machine translation (Caglayan et al., 2016), image (Xu et al., 2015b) and video description (Hori et al., 2017). Broadly, an attention mechanism modifies the output of a sequence representation, based on the coherence of each of the elements (“tokens”) of the sequence to a specific “query”. Information learnt by the attention mechanism may have a distinct conceptual importance (e.g. alignments in machine translation) or simply indicate which elements of the input contribute more to the final output representation.

Attention mechanisms can be trained in an unsupervised way along with the rest of the network in an end-to-end fashion, or can be explicitly supervised by providing the model with pre-calculated attention scores. Supervised attention has been shown to boost the performance of models for machine translation (Mi et al., 2016), constituency parsing (Kamigaito et al., 2017), event detection (Liu et al., 2017b) and aspect-based sentiment analysis (Cheng et al., 2017). Furthermore, image attention mechanisms guided by weak or direct supervision

have been proposed for the tasks of image (Liu et al., 2017a) and video captioning (Yu et al., 2017).

Multi-head attention mechanisms (Vaswani et al., 2017) employ more than one, independent, attention mechanisms, boasting multiple areas of focus on the input sequence. The main idea behind them is that a single attention head may not prove adequate to capture all the different types and positions of information that are important to the end task. We experiment with an attentive model variant that uses supervised multi-head attention to take advantage of annotations in two levels of granularity.

In the model outlined above, each episode is assumed to be divided to snippets, and the sequence model operates and makes predictions on the snippet level. At this level, the input is trimodal: text from the subtitles, audio and video. However, the text modality for each timestep may contain a variable number of tokens. We assume the presence of a text encoder that creates a fixed-length text representation for the text of each snippet. The role of such an encoder can be played by a convolutional sentence encoder (Kim, 2014) or an RNN encoder.

In order to take advantage of attention mechanisms, we conduct a set of experiments using an RNN text encoder, wherein the encoded text representation for each text snippet is weighted by attention scores calculated over its tokens. In our setup, the “query” and “token” representations come from the same sequence, making our mechanism a *self-attention* one (Yang et al., 2016).

More specifically, our attention mechanism works by using the dot product to calculate the attention scores (dot product attention; Luong et al. 2015):

$$\text{score}(s_{ti}, x_t^{(1)}) = x_t^{(1)\top} s_{ti}, \quad (5.4)$$

where  $s_{ti}$  refers to the output of the token-level RNN at timestep  $t$ , word  $i$ ,  $i \in \{1, 2, \dots, k\}$  and  $x_t^{(1)}$  the final encoded state of the textual modality at timestep  $t$ . Given those scores, alignment vectors, which are used as weights to calculate the final representation are calculated as

$$a_t(s) = \frac{\exp(\text{score}(s_{tm}, x_t^{(1)}))}{\sum_{m'} \exp(s_{tm'}, x_t^{(1)})}. \quad (5.5)$$

The CSI dataset includes two levels of annotation for perpetrator mentions: a token level and a sentence level. The two levels of annotation in the dataset follow a compositional structure: the presence of at least one token annotated with 1, results in the whole sentence annotated with 1. In order to train our supervised attention model, we use the token-level annotations,

which we split conceptually to three different types (first person pronoun tokens, pronoun tokens, other type of tokens). Using this split, we are able to train three different attention heads, each focusing on a different type of annotation. Using multiple attention heads leads to multiple sentence representations for the text modality; we use mean pooling to create the final representation that is in turn fed to the utterance-level, multi-view model.

While the architecture of the three heads is identical, we use each of them to model a different type of annotation. For each head, the score generated for each token, is supervised by annotation values of each of the types of annotation. For sentences with more than one tokens are annotated with 1, we equally divide the annotation value between the tokens. As such, we need to modify our learning objective, so that it includes a loss term for each of the heads. We choose the mean squared error:

$$\mathcal{L}_{\text{att}_a} = \sum_t \sum_i (\hat{y}^{(t)} - y^{(t)})^2, \quad (5.6)$$

and the overall objective of the model becomes

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda \mathcal{L}_{\text{corr}} + \lambda_1 \mathcal{L}_{\text{att}_a} + \lambda_2 \mathcal{L}_{\text{att}_b} + \lambda_3 \mathcal{L}_{\text{att}_c}, \quad (5.7)$$

where  $\mathcal{L}_{\text{att}_a}$ ,  $\mathcal{L}_{\text{att}_b}$  and  $\mathcal{L}_{\text{att}_c}$  are the loss terms for heads a, b and c respectively and the parameters  $\lambda_1$ ,  $\lambda_2$  and  $\lambda_3$  weigh the contributions of each attention head to the overall loss.

The multi-head attention component of our model bears similarities in spirit to the recent work of [Strubell et al. \(2018\)](#), where an attention head is replaced by a model trained to predict syntactic dependencies ([Dozat and Manning, 2017](#)). In contrast, our model uses explicit supervision for all self-attention heads and is trained to predict the correct attention scores in a multi-task fashion. A schematic depiction of this model can be seen in [Figure 5.2](#) and the corresponding experiments in [paragraph 5.3.2](#) (Supervised Multi-head Attention).

Inference on multimodal sequences can take the form of inferring a label for a whole sequence or a label for each of the parts of it. The multimodal LSTM of [Ren et al. \(2016\)](#) bears similarities to our work and although it is applied in a sequential inference setting, it does not produce a joint representation for all modalities, but rather, different (albeit informed about each other) modality representations are used for inference.

Our approach is more related to that of [Yang et al. \(2017\)](#), where a multimodal encoder-decoder model for representation learning of temporal data is described. Our method uses their corrGRU cell with a distinct architecture and loss function: first, their network is used as an unsupervised sequence representation learning tool trained to generate an embedding



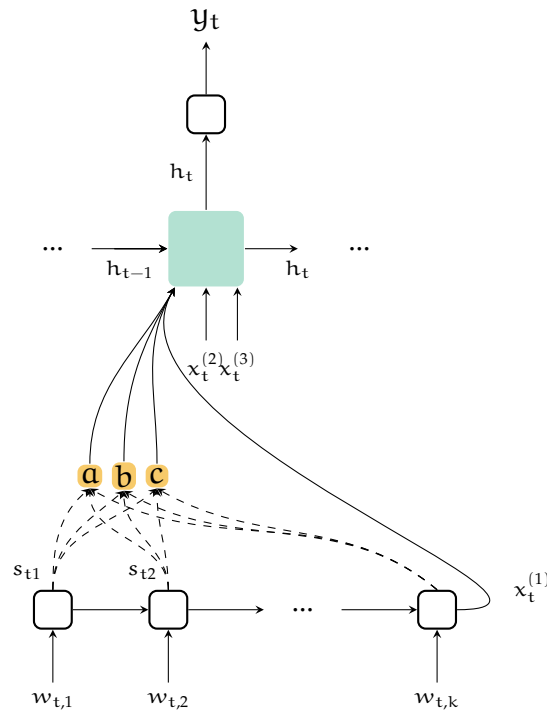


Figure 5.2: Hierarchical multi-view recurrent model with multi-head attention. Each sentence is encoded with an RNN and three attention heads (a, b and c) calculate attention scores for each of the tokens  $w_{ti}$  of the  $t$ -th sentence of the script.

for a whole sequence, which in turn is used for classification tasks, while our model outputs embeddings and makes predictions at the token-level (for every element of a sequence). Secondly, they use a decoder which reconstructs the original representations of each view, while our model does not include an autoencoding part.

Casting correlation maximisation between three or more variables as the maximisation of the sum of the correlation between all pairs of available variables has been previously used in extensions of CCA for more than two views (Benton et al., 2019), or other multi-view learning works (Kumar et al., 2011). Yang et al. (2017) mention it in their paper, although they do not experiment with it.

### 5.3 Experiments

The following section describes experiments conducted in support of the proposed architecture. We use the CSI dataset (Fremann et al., 2018), which consists of 39 episodes of the television series *CSI: Crime Scene Investigation*. In each episode, a team of detectives undertake the solution of one (in 51% of the episodes) or two (49%) crime cases. The three

modalities included are text scripts (dialogue subtitles and background scene descriptions), image snapshots from the video and audio segments. Each sentence of the script is aligned with image and audio<sup>3</sup> excerpts. Also, sentences are annotated with the case they belong to (binary label), perpetrator mention labels (binary) and the name of the speaker that uttered them (“None” for scene descriptions). Each speaker belongs to one of the types *detective*, *perpetrator*, *suspect*, *extra* (“None” for scene descriptions). To facilitate tagging, we convert the case and speaker annotations of the dataset to annotations employing the BIOU (Beginning, Inside, Outside, Unit)<sup>4</sup> format, derived from the BIO scheme proposed for text chunking (Ramshaw and Marcus, 1999) and heavily used in the CoNLL shared tasks<sup>5</sup> for sentence tagging. An annotated example excerpt is shown in Figure 5.3.

### 5.3.1 Experimental Setup

For all experiments, we adopted an experimental setup similar to that of Frermann et al. (2018). For text, we use 50-dimensional pre-trained GloVe vectors (Pennington et al., 2014) and a convolutional text encoder with maxpooling (filters of sizes 3, 4 and 5, each returning a 75-dimensional output). Image features are generated by the final hidden layer of the inception-v4 model (Szegedy et al., 2017) with dimensionality of 1,546. Audio features are constructed by concatenating five 13-dimensional Mel-Frequency Cepstral Coefficient (MFCC) feature vectors for each interval. For perpetrator mention identification, we use the case level splits, whereas the speaker and case experiments are performed on the episode level. All LSTM and GRU variants have one layer of length 128 and a dropout probability of 0.5 is used. We set the value of parameters  $\lambda = 0.001$ ,  $\lambda_a = 0.001$ ,  $\lambda_b = 0.001$ ,  $\lambda_c = 0.001$  and train for 150 epochs with the Adam optimiser (Kingma and Ba, 2014), setting the initial learning rate at 0.001.

### 5.3.2 Perpetrator Mention Identification

In order to investigate the effectiveness of our model in the sequential multimodal inference setup, we conduct the following set of experiments on the task of perpetrator mention identification.

<sup>3</sup>Dialogues have been stripped from the audio track, leaving it only with audio effects and music, so that the text modality will not be deemed redundant and the dataset does not contain overlapping information.

<sup>4</sup>The BIOU is a subset of the BILOU tagging format (Ratinov and Roth, 2009). In the BIOU format, each of the elements of a sequence is tagged with a label that indicates the type of chunk it belongs to and its position within the chunk. Specifically, B: Beginning (marks the beginning of a chunk), I: Inside (denotes an element that is inside a chunk), O: Outside (the element does not belong to any chunk) and U: Unit (the element is the first and last element of a chunk).

<sup>5</sup><http://www.conll.org/previous-tasks>



	
NONE	GRISSOM
Grissom doesn't look worried. He takes his gloves off and puts them on the table.	You ever been to the theater, <b>Peter</b> ?
CASE: 1	CASE: 1
PERP: 0	PERP: 1
CASE TAG: B-1	CASE TAG: I-1
SPEAKER TYPE TAG: O	SPEAKER TYPE TAG: B-D

Figure 5.3: An excerpt of the CSI dataset, where the image and text modalities and annotations are shown. The case and speaker tags use the BIOU (Beginning, Inside, Outside, Unit) format. In this case, "Peter" is the name of the perpetrator (Perp: 1). Both snapshots belong to the first case. The first snapshot does not have a speaker (screenplay description; speaker "None" and speaker type tag "O") and starts a chunk of utterances belonging to the first case (B-1). The second snapshot continues in the first case (I-1) and starts a chunk where the speaker is a detective (B-D).

**Multi-view Model** The effectiveness of different architectures is shown in the first section of Table 5.1. The multi-view model (using corrGRU) is compared to early fusion models (using LSTM and GRU cells), for which the input is the concatenation of the feature vectors of the three modalities, passed through a ReLU activation. It can be seen that the multi-view model outperforms all other models.

**Contribution of cell components** The correlational GRU cell includes a set of distinctive components: (a) a dynamic weighting module and (b) a correlation loss term. In order to assess the contribution of each of those features, we conduct an ablation experiment: we train our model by removing the correlation term from the loss (setting  $\lambda$  to 0), by removing the dynamic weighting module (setting all the weights to 1) and by removing both. The results of this ablation are outlined in Table 5.2. We conclude that both the weighting and correlation modules are important for the final prediction.

**Incremental Inference** *Incremental sequence labeling* refers to making predictions on an incoming sequence as it is “streamed” in an online fashion. For example, if the sequence is a sentence, we are not allowed to encode the whole sentence first, but instead have to output a relevant label for each word as it arrives in the sequence. Incrementality underlies fundamental human cognition and is essential for scaling systems to large datasets and real-time inference, necessary, for example, in simultaneous translation (interpretation; [Bangalore et al. 2012](#); [Yarmohammadi et al. 2013](#); [Cho and Esipova 2016](#)).

In order to assess the effectiveness of the incremental inference capabilities of our model, we contrast the output of incremental models (forward-pass unidirectional) to that of similar models that do not perform incremental inference (bidirectional), in Table 5.1. Both multi-view and non-multi-view bidirectional models look ahead in the sequence, gathering information that is potentially useful for temporal inference. The bidirectional correlational model was trained with an extra correlation loss term, calculated exactly as  $\mathcal{L}_{\text{corr}}$ , with data from the backward pass. The bidirectional multi-view model does not score as high as the unidirectional multi-view one, though it balances better between precision and recall. Interestingly, results suggest that the incremental multi-view model outperforms early fusion bidirectional models (biLSTM and biGRU).

	model	pr	re	F1
UNI-DIR	EF (LSTM)	42.8	51.2	46.6
	EF (GRU)	39.4	60.4	47.7
	MV (corrGRU)	41.3	<b>63.4</b>	<b>50.0</b>
BI-DIR	EF (biLSTM)	40.0	62.7	48.8
	EF (biGRU)	43.6	58.1	49.8
	MV (biCorrGRU)	<b>49.6</b>	49.4	49.5

Table 5.1: Comparing unidirectional and bidirectional variants of early fusion models and our multi-view model. Precision (pr), recall (re) and F1 scores for detecting the minority class (perpetrator mentioned) are reported on the held-out dataset. EF stands for early-fusion, while MV for our multi-view model. The first section of the table reports scores for unidirectional (incremental) models and the second for bidirectional (non-incremental) models. The result for the simple unidirectional LSTM model is the one reported by [Frermann et al. \(2018\)](#).

model	pr	re	F1
full model	41.3	63.4	50.0
no DW	38.9	58.3	46.7
$\lambda = 0$	39.0	59.2	47.0
$\lambda = 0$ , no DW	37.5	58.3	45.6

Table 5.2: Assessing the contribution of the components of our model. Precision (pr), recall (re) and F1 scores for detecting the minority class (perpetrator mentioned) are reported on the held-out part of the dataset. The setups compared are: the full model, the model without the dynamic weighting (DW) module, the model without the correlation loss term ( $\lambda = 0$ ) and the model without both the dynamic weighting and the correlation loss.

**Contribution of the different modalities** We conduct an ablation experiment assessing the contribution of each modality to the final prediction. The results of this experiment can be found in Table 5.3. Evidently, all three modalities contribute to the good performance of the multi-view model. We note that the *text* modality is the most informative; models taking text into account score consistently better, in both multi-view and single view setups. Results of the single-view video model suggest that the *image* modality alone provides very little information about the perpetrator’s identity. It is possible that the general nature of the features used are partially responsible for that, since the `inception` model is trained on object recognition and not a face recognition task. Results on *audio* only are not reported, since the audio modality contains only music and audio effects and is not expected to generate useful representations by itself.

**Supervised Multi-head Attention** The CSI dataset includes token-level perpetrator mention annotations: every token of the script sentences is tagged as being a mention of the perpetrator or not. An example can be found in Figure 5.3 (right), where “Peter” is tagged as being a reference to the perpetrator (boldface). The sentence-level annotations used throughout the previous experimental section and throughout the work of [Fermann et al. \(2018\)](#) are generated by aggregating token-level annotations; the token-level annotations are not used in their work though.

We distinguish three types of perpetrator token mentions in the dataset:

- *first person pronoun tokens*: the perpetrator is the speaker and speaks in first person.
- *pronoun tokens*: other characters refer to the perpetrator by using pronouns.

	MODALITY			pr	re	F <sub>1</sub>
	T	I	A			
THREE	✓	✓	✓	41.3	<b>63.4</b>	<b>50.0</b>
	✓	✓		<b>41.4</b>	49.6	45.1
TWO	✓		✓	39.6	50.4	44.3
		✓	✓	38.7	5.1	9.0
ONE	✓			41.9	47.3	44.4
		✓		28.4	6.7	10.8

Table 5.3: Ablation experiment assessing the contribution of each modality in our multi-view model. Precision (pr), recall (re) and F<sub>1</sub> scores for detecting the minority class (perpetrator mentioned) on the held-out part of the dataset are reported. The modalities are denoted by T (text), I (image) and A (audio). For single-view setups, we report results only using the Image and Text modalities, since the Audio modality is not quite informative on its own.

- *other type of tokens*: perpetrator is mentioned by their name or other attributes.

We replace the original binary token-level annotation with three binary annotation streams reflecting the three different types (first person pronoun mention, other person pronoun, other type of mention).

We replace the convolutional encoder of the previous experimental setup with an LSTM and three attention heads and run experiments comparing attentive architectures with non-attentive ones. The results can be found in Table 5.4. Unsurprisingly, models that make use of the extra information in the form of attention supervision score better than their counterpart that does not take token-level annotations into account. Interestingly, the complexity and diversity of token-level annotations is reflected in the results of single-head attention models: the supervised single-head model scores lower than the one that is free to learn any attention scores. Ultimately, the choice to split the annotations to three streams and use more than one attention heads, each focusing on different types of mentions, leads to better performance.

However, even the multi-head supervised attentive model, does not score as well as the multi-view (non-attentive) model. This result highlights the, sometimes disregarded, importance of modality fusion: creating a fused representation out of the available modalities led to a model that outperforms one with significantly more information in its disposal.

model	pr	re	F1
EF	<b>42.8</b>	51.2	46.6
MV	41.3	<b>63.4</b>	<b>50.0</b>
EF+ATT	39.95	58.70	47.32
EF+SUPATT	40.72	56.11	47.15
EF+MULTIHEAD	40.25	59.19	47.88

Table 5.4: Comparing the performance of early fusion (EF) and multi-view (MV) models with attentive early fusion models. Three different attention schemes are compared: simple attention (ATT), supervised attention (SUPATT) where the network’s loss includes an error term for the attention scores with respect to the token-level annotations, and multi-head supervised attention (MULTIHEAD) where the token-level annotations are divided conceptually into three groups and each head is supervised by the scores of one of the groups.

### 5.3.3 Episode Structure Tagging

Extracting knowledge about a movie by relying on simplified tasks can be challenging and may require assumptions about the input. Specifically, casting the task of perpetrator identification as a binary classification task, is based on the premise that there is, for every input, at most one perpetrator. This assumption does not always hold, since, some episodes contain two cases and consequently, two perpetrators. Moreover, new perpetrators are introduced in every episode and data sparsity makes multi-class classification difficult. For the experiments described in the previous sections, we alleviate this obstacle by performing binary inference on the case level, using the annotated case splits of the dataset.

In order to enable more robust movie understanding, we investigate the automatic segmentation of episodes to coherent chunks by experimenting with tagging utterances with tags of two levels of granularity: case and speaker type. The former refers to associating each utterance with the crime case it belongs to, while the latter to labeling each utterance as coming from one type of speaker (detectives, perpetrators, suspects, extras) or none (for scene descriptions).

The two tagging tasks are closely related, since a shift from a speaker type (e.g. a conversation between detectives) to another (e.g. a conversation between extras) may indicate a shift in the focus of the episode, hinting a case change. The presence of more than one related tasks makes our setup ideal for testing our model in a multi-task setting. Sharing representations between tasks is justified by the notion that information from similar tasks can aid in solving

model	SPEAKER TYPE				CASE			
	acc	pr	re	F1	acc	pr	re	F1
EF	50.55	20.28	24.18	20.66	61.00	0.01	0.01	0.01
MV	57.66	19.87	35.95	25.29	61.65	0.03	2.86	0.05
EF+CRF	49.70	15.70	14.22	14.48	62.75	3.10	15.21	4.97
MV+CRF	51.27	14.89	16.71	14.96	73.53	<b>11.72</b>	27.75	11.24
MULTI-TASK (SPEAKER+CASE)								
EF	45.07	19.83	27.53	21.82	61.75	0.00	0.00	0.00
MV	46.11	18.17	22.36	19.09	61.02	0.06	0.08	0.06
EF+CRF	47.04	<b>22.02</b>	17.42	18.60	73.95	1.09	1.52	1.11
MV+CRF	<b>60.07</b>	21.36	<b>39.25</b>	<b>27.61</b>	<b>79.49</b>	8.29	<b>44.17</b>	<b>13.72</b>

Table 5.5: Performance of early fusion (EF) and multi-view (MV) model variants on speaker type and case tagging. Macro-average scores for accuracy (acc), precision (pr), recall (re) and F1 scores are reported. The top section of the table refers to the single-task setup, while the bottom on the multi-task setup (training jointly on speaker type and case tagging).

the task at hand faster and more accurately (Caruana, 1997).

We modify the architecture of our model, so that the output of the sequence model cell is fed to different output layers, one for each task. Training proceeds by summing the loss terms for both tasks. In the case of our multi-view model, the loss consists of two cross-entropy terms and one correlation term. Moreover, we experiment with adding a Conditional Random Field (CRF) on top of the sequence models, based on recent work that achieves state-of-the-art performance in tagging tasks, such as Named Entity Recognition (Lample et al., 2016).

The results for speaker type tagging can be found in Table 5.5, where our model (MV) is compared with an LSTM early fusion model. We use a variant of the evaluation script used for the CoNLL shared tasks<sup>6</sup> and report average scores. It can be seen that our multi-view model consistently outperforms early fusion models. Interestingly, the multi-task MV+CRF model exhibits the best performance, suggesting that jointly solving the two tasks improves the capabilities of the model.

<sup>6</sup><https://github.com/spyysalo/conlleva1.py>



## 5.4 Conclusions

In this chapter, we describe a neural multi-view sequential architecture, paired with a novel objective that takes advantage of supervision, while at the same time, maximises the correlation between views. We test our approach on the task of perpetrator mention identification of the CSI dataset, on which we show that it outperforms state of the art. Also, we introduce two shallow movie understanding tasks, crime case and speaker type tagging, and show that our model yields consistently better results than early fusion models, highlighting the importance of careful fusion of modalities in sequential inference.

# Chapter 6

## Narration Generation from Video Input

*we pose the question: “Is photorealism necessary for the study of semantic understanding?” [...] cartoons or comics are highly effective at conveying semantic information without portraying a photorealistic scene.*

*(Zitnick and Parikh, 2013)*

Text generation from visual or multimodal inputs has been an overarching goal and a point of convergence of the Computer Vision and Natural Language Processing communities (Gatt and Kraemer, 2018). Research in this direction has been propelled by the proposal and study of several tasks, with examples including image caption generation (see Bernardi et al. 2016 for a survey), visual question generation (Mostafazadeh et al., 2016), caption explanation (Hendricks et al., 2016), visual question answering explanation (Li et al., 2018) and multimodal machine translation (Calixto et al., 2017; Lala and Specia, 2018). Progress in corresponding video tasks, such as video description, video question answering (Zeng et al., 2017; Xu et al., 2017; Tapaswi et al., 2016) and video overview generation (Gorinski and Lapata, 2018) has been slower, probably due to the extra challenge posed by the temporal nature of videos. In this chapter, we present *narration generation*, a new text generation task from movie videos. We believe that the introduction of this task will challenge existing techniques for text generation from videos as it requires temporal contextual reasoning and inference on storylines.

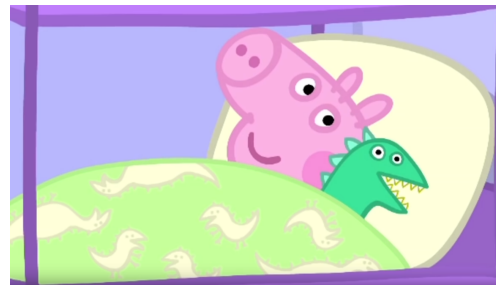
A narration is a commentary commonly used in movies, series or books. It can be delivered by a story character, a non-personal voice or the author of a book and may communicate a story that is parallel to the plot, fill in details that are not directly perceivable, or help in



(1a)

01:19 – 01:20

MOMMY PIG: Goodnight, George.



(1b)

01:24 – 01:27

NARRATOR: When George goes to bed, Mr Dinosaur is tucked up with him.



(2a)

03:29 – 03:31

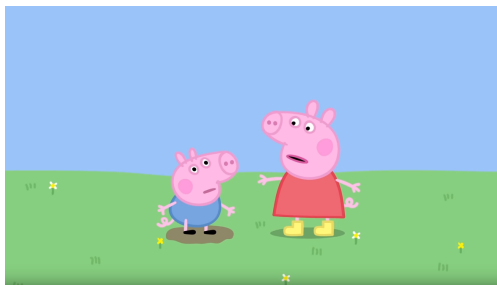
PEPPA PIG: See, that's where it is.



(2b)

03:32 – 03:34

NARRATOR: Mr Dinosaur is not in George's bed.



(3a)

01:20 – 01:26

PEPPA PIG: George, if you jump in muddy puddles, you must wear your boots.



(3b)

01:29 - 01:34

NARRATOR: Peppa likes to look after her little brother, George.

Figure 6.1: An excerpt of the Peppa Pig dataset. The first two examples (rows) are from *Episode 2: Mr Dinosaur is lost*, while the last from *Episode 1: Muddy Puddles*. For each subtitle, a representative snapshot (image) is shown. It can be seen that the narrations may or may not be descriptive of the image (or short clip) which they accompany.

guiding the viewers/readers through the plot. *Narration generation* refers to the task of automatically generating the text of such narrations. Our focus is on video data, especially videos that are part of episodic broadcasts, such as television series.

To facilitate research in the direction of automatic narration generation, we create a new narration dataset. Following the spirit of previous work on image captioning from abstract scenes (Zitnick and Parikh, 2013) and cartoon video question answering (Kim et al., 2016), we collect videos from the animated series *Peppa Pig*. We posit that abstracting away from related, but nonetheless hard problems, such as processing real-life photographs and videos and understanding complex, real-life dialogue between adults, will make for clean and isolated evaluation of the text generation techniques themselves.

Narration generation in the way we set it up, is a new task, distinct from video description in certain aspects. Compared to descriptions, narrations provide high-level, less grounded information on events taking place in videos. In general, they do not articulate objects or actions that can be directly seen in the images and even in cases where they do, the description is quite specific and tied to the context of the overall story. For example, the scene of Figure 6.1(1b), could be accurately described by “a pig in a bed, with a toy tucked up with it”. The narration, however is context-aware: it refers to the pig as “George” and the toy as “Mr Dinosaur”.

Moreover, a narration may refer to events or objects that cannot be seen in the image. Image captioning algorithms face striking challenges when it comes to objects absent from query images, which can be easily inferred by humans (Bernardi et al., 2016), such as the “bus” in the caption of an image showing people waiting for a bus on a bench. Narrations may include contextual mentions to absent objects, such as the reference to “Mr Dinosaur” in Figure 6.1(2b); something that a viewer not familiar with the storyline could not have guessed. Finally, narrations may convey information less related to their accompanying video and more to the overall plot: Figure 6.1(3b), for example, makes a remark on Peppa looking after her brother, while the image just shows Peppa near a puddle.

Video narration generation bears resemblance to the task of automatic generation of sports broadcasts, known as sportscasts (Chen and Mooney, 2008). While sportscasts can have narrative structure (Herman et al., 2010), they are generated on the fly, and contain information related to what has been shown in the video up to the point of their utterance. Conversely, the narrations of our dataset are third-person omniscient narrations: the narrator knows everything related to the storyline and may use information that is being shown to the viewers



Figure 6.2: A snapshot from a news summary video which includes a form of narration (text shown between video excerpts to fill in gaps in the story). From the New York Times website.<sup>1</sup>

while the narration is uttered or use forward references to events that are to unfold later in the video. In terms of content, sports commentary that does not simply describe the game, known as “color commentary” (Lee et al., 2014) is more relevant to movie narrations.

Our work is a step towards the direction of narrative content generation and serves as a proxy problem for several applications. Examples include not only sports commentary, but also other types of commentary (such as director’s commentary in movies) or content for news summary videos (see Figure 6.2).

The contributions of this chapter are as follows:

- We introduce and formalise the task of narration generation from videos.
- We develop a new cartoon video dataset for the task of narration generation.
- We present several models for narration generation and report results on the newly introduced dataset.

## 6.1 The Peppa Pig Dataset

*Peppa Pig* is a popular British animated television series targeting preschool children. The episodes follow Peppa, a young female pig, in everyday social, family, and school activities. Main characters also include Peppa’s younger brother (George), their mother (Mommy Pig),

<sup>1</sup><https://www.nytimes.com/2019/05/01/us/politics/william-barr-testimony.html>, accessed 2 May 2019, 14:30.

episodes	209
total time	1045 min
time excluding intro & outro	927 min
narrations	1803
dialogue length (avg)	56.9 tokens
narration length (avg)	10.7 tokens
narration vocabulary	1771 words
narration unique vocabulary	257 words

Table 6.1: Statistics of the Peppa Pig dataset.

father (Daddy Pig) and various friends and relatives. All the friends of Peppa’s family belong to different animal species than pigs. Although all the characters of the show are animals, most of them exhibit human traits and lead human-like lives (they speak human languages, wear clothes, live in houses, drive etc.). At the same time, they keep some animal features, such as the distinctive sound of their species (pigs, for example, snort during conversations).

The simplicity of images, dialogues and storylines of Peppa Pig, make it an ideal testbed for movie understanding experimentation. The small scale of the vocabulary and topics alleviate sparsity challenges that can potentially arise in, relatively small but diverse, movie datasets. Finally, the fact that the storyline of each episode includes a narrator, makes the task of eliciting narration data relatively easy.

The series first aired on 2014 and as of the end of 2018, 264 episodes have been created. Our dataset consists of 209 episodes, for which we were able to find subtitles online.<sup>2</sup> A full list of the episodes used for the dataset can be found in Appendix B.1. For each episode, we collect the video file, subtitles and metadata (title, plot summary, air date). Some descriptive statistics of our dataset can be found in Table 6.1, while Figure 6.1 lists three example scenes taken from the first two episodes of the series. More details about narration and dialogue lengths, number and positions in episodes can be found in Figure 6.3.

A preprocessing step was carried out to make sure that all the videos are stripped out of unnecessary parts (each episode starts with the main character introducing herself and her family and usually ends with a song), subtitles are synchronised with the video, subtitle text is normalised, and each token of the text is aligned with a corresponding timeframe (a process called “forced alignment”). The resulting dataset is aligned at the token level, for image,

<sup>2</sup>We make our dataset available online: [https://github.com/papagandalf/peppa\\_pig\\_data](https://github.com/papagandalf/peppa_pig_data).

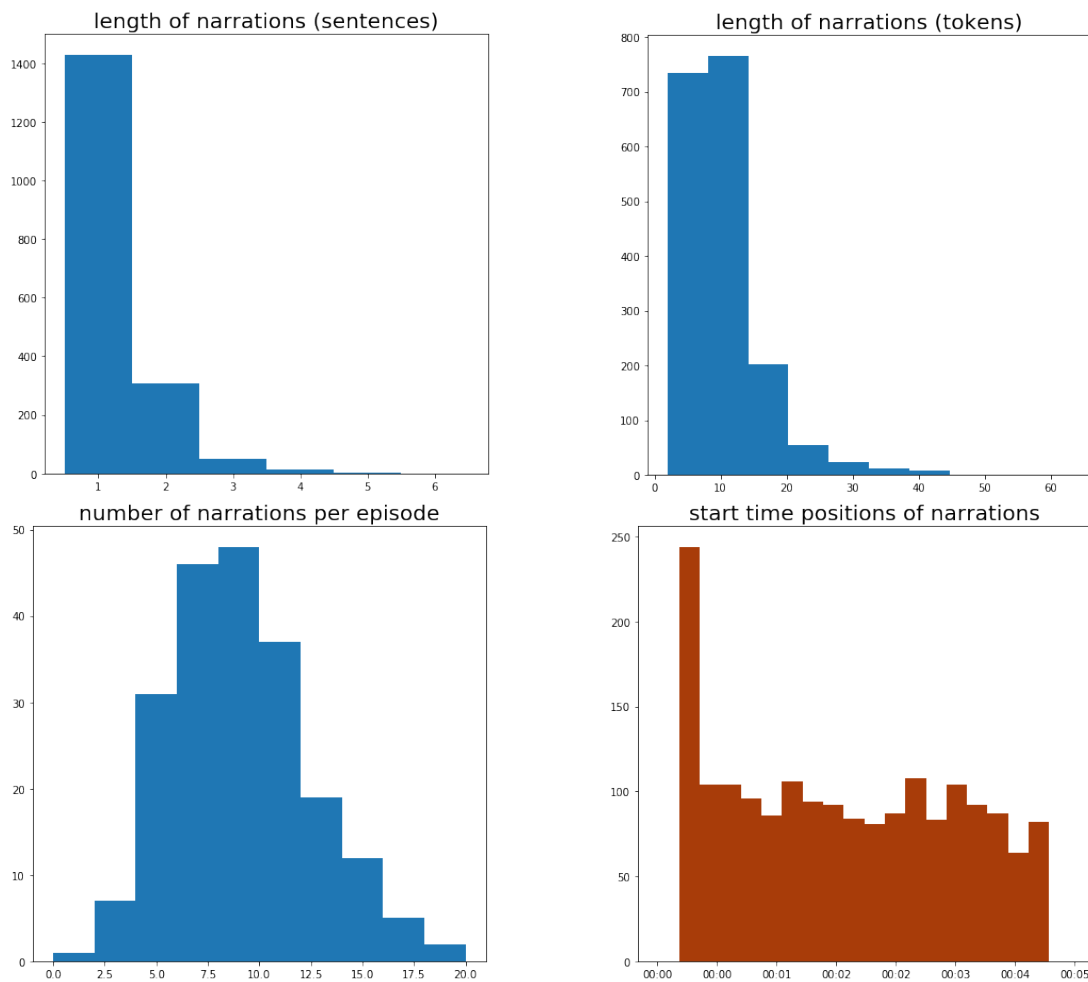


Figure 6.3: Histograms of characteristics of Peppa Pig narrations. The length of narrations in sentences and tokens, the start time position within the episode and the number of narrations per episode are listed.

audio and text modalities. Detailed account of the preprocessing steps can be found in Appendix B.2.

**Features** For each episode, we calculate feature vectors for the image and audio modalities, as follows. For image frames, we use ResNet-50 (Residual Networks; He et al. 2016) and VGG-19 (from the the work of Visual Geometry Group; Simonyan and Zisserman 2015) to get object recognition features. For audio excerpts, after stripping the dialogues from the audio track,<sup>3</sup> we calculate Mel-Frequency Cepstral Coefficients (MFCC; Mermelstein 1976), commonly used as features in speech (Gupta et al., 2018) and music (Jensen et al., 2006) signal

<sup>3</sup>Information on the content of the dialogues is already present in the text modality. Even after stripping speech out, audio can be informative since it contains sound effects and music.

dataset	domain	task	videos	avg length (sec)	vocab	length (min)
MSVD (Chen and Dolan, 2011)	open	description	1,970	10	13,010	318
MPII Cooking (Rohrbach et al., 2012)	cooking	description	44	600	-	480
YouCook (Das et al., 2013)	cooking	description	88	-	2,711	138
TACoS (Regneri et al., 2013)	cooking	description	127	360	28,292	954
TACoS-ML (Rohrbach et al., 2014)	cooking	description	185	360	-	1,626
MPII-MD (Rohrbach et al., 2015)	movie	description	94	3.9	24,549	4,416
M-VAD (Torabi et al., 2015)	movie	description	92	6.2	17,609	5,076
MovieQA (Tapaswi et al., 2016)	movie	QA	408	7,200	-	-
PororoQA (Kim et al., 2016)	cartoon	QA	171	432	3,729	1,230
Peppa Pig	cartoon	narration	209	300	1,771	1,045

Table 6.2: Multimodal datasets for tasks related to text generation from videos.

applications. Additionally, we calculate VGGish<sup>4</sup> features (Hershey et al., 2017; Gemmeke et al., 2017), which are reported to produce state-of-the-art results in audio classification and have also been used for video description (Hori et al., 2018).

**Comparison with Other Datasets** Our dataset adds to the set of existing datasets for multimodal video understanding. Table 6.2 lists features of several datasets pertaining to text generation from videos, such as video description and video question answering. The Peppa Pig dataset is the first dataset on narration generation.

### 6.1.1 Narration as Video Summary

Besides their apparent purpose in books and videos, narrations can also serve as a form of video summary. Consider for example, narrations from two episodes of Peppa Pig along with their corresponding plot summaries (from IMDb)<sup>5</sup> and plot sentences (from Wikipedia),<sup>6</sup> shown in Figure 6.4. Clearly narrations are more elaborate than plot summaries, but do not fail to convey the main events taking place during the episode. Narrations and plot summaries can both be regarded summaries of the video, in different layers of abstraction.

<sup>4</sup>This is the name of the pre-trained model we used, available from <https://github.com/tensorflow/models/tree/master/research/audioset>.

<sup>5</sup>[www.imdb.com](http://www.imdb.com)

<sup>6</sup>[en.wikipedia.org/wiki/List\\_of\\_Peppa\\_Pig\\_episodes](http://en.wikipedia.org/wiki/List_of_Peppa_Pig_episodes), not available for all episodes.



narration	narration
<p>It is raining today, so Peppa and George cannot play outside.</p> <p>Peppa loves jumping in muddy puddles.</p> <p>George likes to jump in muddy puddles, too.</p> <p>Peppa likes to look after her little brother, George.</p> <p>Peppa and George are having a lot of fun.</p> <p>Peppa has found a little puddle.</p> <p>George has found a big puddle.</p> <p>George wants to jump into the big puddle first.</p> <p>Peppa and George love jumping in muddy puddles.</p> <p>Peppa and George are wearing their boots.</p> <p>Mummy and Daddy are wearing their boots.</p> <p>Peppa loved jumping up and down in muddy puddles.</p> <p>Everyone loves jumping up and down in muddy puddles.</p>	<p>Mommy Pig is working on her computer.</p> <p>Daddy Pig is making soup for lunch.</p> <p>Mommy Pig has a lot of important work to do.</p> <p>Peppa and George love to watch Mommy work on the computer.</p> <p>Oh, dear, the computer is not meant to do that.</p> <p>Daddy Pig is going to mend the computer.</p> <p>Daddy Pig has mended the computer.</p>
plot summary	plot summary
<p>It is raining and Peppa is sad because she cannot go outside. When the rain stops, Peppa and George get to play one of their favourite games - jumping in muddy puddles. Things get very muddy indeed when Mummy and Daddy Pig join in.</p>	<p>Peppa and George accidentally break Mummy Pig's computer, so Daddy Pig tries to fix it.</p>
plot	plot
<p>Peppa and George get very muddy after playing their favourite game - Muddy Puddles.</p>	<p>Peppa breaks Mummy Pig's computer while she is working.</p>

Figure 6.4: Narrations and plot summaries from two episodes, *Episode 1: Muddy Puddles* and *Episode 7: Mummy Pig at Work*.

metric	75 bytes			full length		
	prec	rec	F1	prec	rec	F1
ROUGE-1	19.98	19.40	19.65	31.74	11.88	16.42
ROUGE-2	5.61	5.48	5.53	6.27	2.37	3.23
ROUGE-L	15.32	17.87	16.47	28.22	10.55	14.58

Table 6.3: ROUGE scores (precision, recall and F1) comparing the plot summaries to the corresponding narration sentences of each episode.

To further explore this direction, we automatically compare (using ROUGE; Lin 2004) plot summaries and narrations of the episodes of our dataset, in order to assess the capabilities of an oracle narrator model as a summariser. Table 6.3 reports ROUGE scores for full length summaries and limited length summaries (at 75 bytes).<sup>7</sup> Interestingly, the limited length scores are relatively high, pointing to the fact that the first narration sentence of each episode contains important information for its summary. However, the overall low scores demonstrate that narration generation is quite a distinct task from summarisation, and points to the value of a dataset for this problem.

## 6.2 Narration Generation

We formalise the task of automatic narration generation as follows. Assuming a set of  $M$  videos, for which  $N$  modalities are available, we regard each video as a sequence of  $T$  elements  $x_{ji}^k$ ,  $j \in \{1, 2, \dots, M\}$ ,  $i \in \{1, 2, \dots, T\}$ ,  $k \in \{1, 2, \dots, N\}$ , where  $k$  indexes the available modalities. The task is to identify which elements should belong to narration and generate appropriate text for them.

The segmentation in  $T$  elements can be done in three different levels: dialogue-narration (D/N), token, and time. The first type splits the video in points where the dialogue ends and the narration begins and vice versa.<sup>8</sup> The second type splits the video in every token of the dialogue, and the third is a time scale, meaning that the video can be split in any timestamp.

We divide the task of narration generation in two separate tasks, *timing* and *content generation*, each solving a particular challenge related to it.

<sup>7</sup>The evaluation was done using `pyrouge`, available in <https://github.com/bheinzerling/pyrouge>.

<sup>8</sup>We refer to the non-narration parts of the video as dialogues, even if they do not necessarily contain dialogues.

### 6.2.1 Narration Timing

Narration timing refers to the task of figuring *when* to place narrations in a video. Depending on the type of input data, a narration may interrupt the flow of speech in the video (as is the case with Peppa Pig: while the narrator speaks, the characters do not engage in dialogue) or it may be superimposed to the rest of the speech (in sportscasts, narrations overlap with other video sounds). In the former case, finding the timing can be regarded as an easier task, since a pause in dialogue can pinpoint the beginning of a narration.

We model and experiment with the more general of the two, this is the reason why we cast the problem of narration timing as an incremental tagging task, where each time step is tagged with a label indicating whether narration follows in the sequence. Incremental sequential models are models that do not have look-ahead capability, that is, they make predictions at each time step using information only from the previous and the current time steps. This is a common setup not only in language modeling, but also in real-time (Cho and Esipova, 2016) or multimodal reasoning applications (Fermann et al., 2018). The constraint of incrementality, which can be satisfied using a simple, unidirectional sequence model, such as a simple LSTM, is important, since a look-ahead model will allow information flow from future nodes, hinting the existence of narration.

Specifically, for this task, we use the token-level segmentation of the dataset and feed an incremental sequential model with multimodal representations (tokens along with corresponding image and audio features). Each token is annotated with a binary label, indicating whether there is at least one narration token in a window of  $n$  tokens right after it. The obvious choice is  $n = 1$ , where each tag indicates whether the immediate next token belongs to narration. Additionally, inspired by work on speech dialogue turn-taking (Skantze, 2017), where  $n > 1$  is used, we create annotations with  $n = 5$ . We refer to the two annotation schemes as *Timing@1* and *Timing@5*. Figure 6.5 contains an annotation example from the Peppa Pig dataset.

Predicting the presence of narration in upcoming time steps is in fact a proxy of the timing problem, hence offering an upper bound on it. The reason for that is that all the aforementioned timing models operate on sequences defined by a pre-calculated quantisation of the video stream on token limits. In the more general case, where the narration part is not given, a more arbitrary chunking of the video has to be used (e.g. every 5 ms). Using the actual narration text (and consequently the correct tokenisation, which provides correct offsets when extracting images and audio excerpts to feed to the model) may make the job of the model

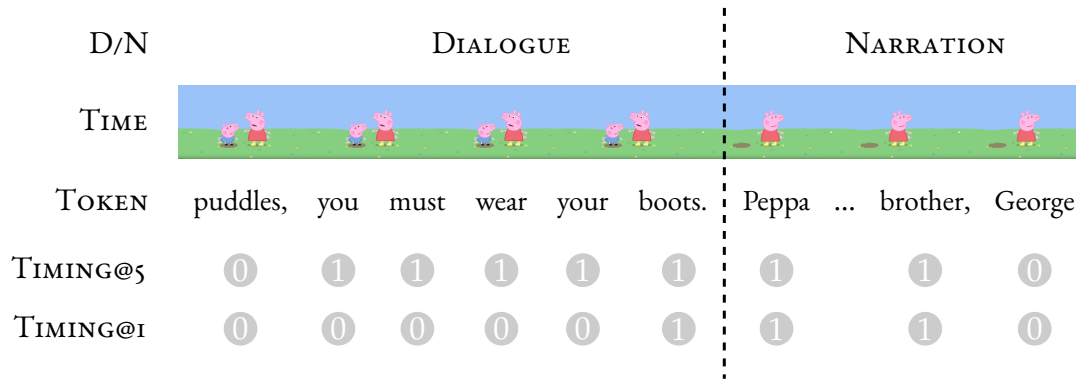


Figure 6.5: An excerpt of an episode of the Peppa Pig dataset, where all the levels of segmentation and tagging schemes for narration timing are shown. *Timing@n* refers to the annotation scheme, where binary labels indicate whether narration is present in a *n*-token window right after each token.

easier than it actually is in the general case.

### 6.2.2 Content Generation

Content Generation refers to the task of figuring *what* to include in the narrations. In order to deal with content generation, we assume that timing is already solved and that videos are correctly segmented into chunks of dialogue and narration.

We hypothesise that a human assigned with the task of coming up with a good narration for a specific part of a video would need to have access to information from several sources. The content of the dialogue preceding the narration is of utmost importance. Equally important are the actions or events taking place in the part of the video that is to be narrated. A narration generator model should have access to the same information, hence we propose a model with the following features:

- takes into account the output of a multimodal dialogue encoder, which encodes dialogue data. We instantiate an LSTM encoder at the token level, which combines information from text, image and audio.
- takes into account the output of a video encoder, which encodes the part of the video to be narrated. Since the corresponding text is the desired output of our decoder, this is a video only (image and audio) decoder. Note that this encoder in principle does not have access to the narration tokenisation, so it should not use the token segmentation, but a time segmentation (e.g. segment every 5ms).

A schematic overview of our proposed narrator model can be seen in Figure 6.6(a). We call this model *Dialogue Video Narrator (DiViNa)*.

**Multimodal representation** Efficient fusion of representations from different modalities in one, multimodal representation is an open research problem. The choice of fusion method is reported to have significant impact on performance on downstream applications (Baltrušaitis et al., 2019). Broadly, fusion techniques can be categorised as *early* (concatenation of representations at the feature level), *late* (concatenation of the output of different modality-specific modules) and *hybrid fusion* methods (Atrey et al., 2010). In order to establish baselines, we experiment with simple early fusion models. Moreover, we report results for multi-view variants of our models, that use the correlational GRU cell (Yang et al. 2017; for a detailed description, see Section 2.3.4) instead of simple LSTMs.

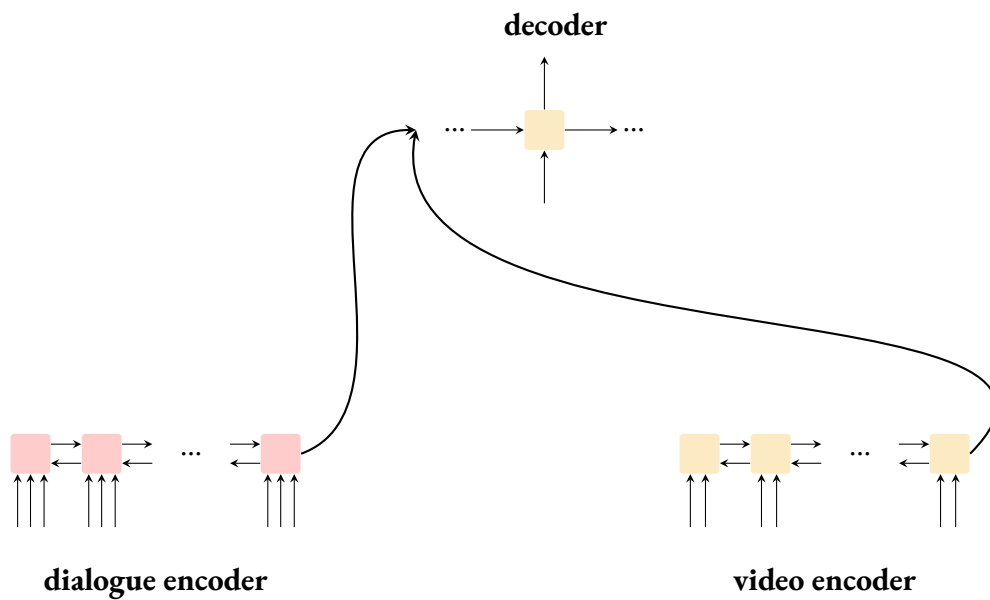
## 6.3 Experiments

This section describes our experiments for both narration timing and content generation. This being a new task, we present a set of baseline models and report their effectiveness.

### 6.3.1 Experimental Setup

For all models described, we use a similar experimental setup. LSTMs with one layer of hidden size 500 are used in places where sequence models are needed. We use pre-trained GloVe embeddings (Pennington et al., 2014) of size 300 for the text modality. The input of multimodal modules is the concatenation of the representations of the different modalities, passed through a linear layer and a ReLU activation. The output size of this linear layer is 300. We train the models using the Adam optimiser (Kingma and Ba, 2014), setting the initial learning rate to 0.001. During narration content generation training, we use teacher forcing (Williams and Zipser, 1989) with probability 0.5. Generation is performed using a beam search decoder, with beam size of 3.

All multimodal models for which scores are reported, use the ResNet-50 features for image and the concatenation of VGG-ish and MFCC features for audio. We decided on this particular combination after preliminary examination of all combinations of the features of the dataset.



(a) Dialogue Video Narrator (DiViNa) model.

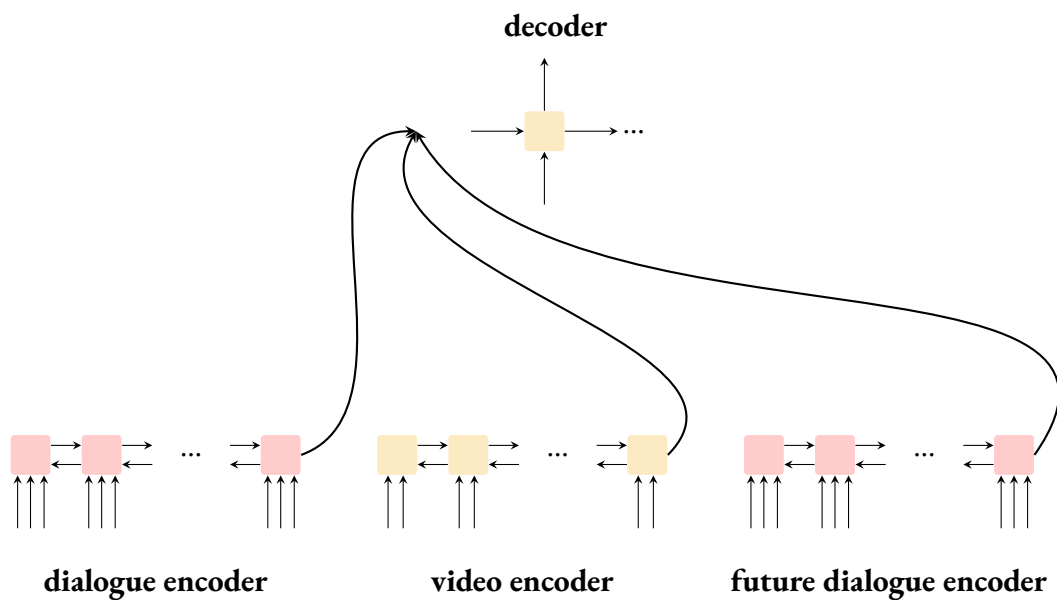
(b) Dialogue Video Narrator model with future dialogue encoder (Di<sup>2</sup>ViNa).

Figure 6.6: Dialogue Video Narrator (DiViNa) and Dialogue Video Narrator with future dialogue encoding (Di<sup>2</sup>ViNa) models. Red nodes operate on dialogue, while yellow ones on the part of the video to be narrated. The input to the previous and future dialogue encoders is trimodal (text, audio, image), hence the three arrows, while the input to the video encoder is bimodal (image and audio only). The outputs of the two or three encoders are concatenated and used to initialise the decoder. The output of the decoder is the narration text.

### 6.3.2 Experiments on Narration Timing

We conduct experiments on timing by training an incremental sequence tagging model, using the Timing@1 and Timing@5 annotations. We experiment with one unimodal (where only the text is given) and two multimodal (text, audio and video) models, one that uses early fusion, and one multi-view model. The evaluation of the output of timing models is done using precision, recall and F1, as is appropriate for a tagging task. We train until the performance on the validation set stops increasing and report results on the test set.

**Implementation details** For our experiments we use the following setup:

- All modalities are passed through a linear layer that yields embeddings of size 300.
- For the multi-view models, the 300 dimensional vectors are passed to the sequence model. For the early fusion model, the representations are concatenated; consequently, the input to the sequence model is of size 900.
- The hidden size of the LSTM is 500.
- We use one-layer sequence models for all experiments.
- Optimisation is done using the Adam optimiser (Kingma and Ba, 2014), with an initial learning rate of 0.001.
- Training proceeds for 100 epochs.
- The model was implemented in PyTorch<sup>9</sup> version 0.4.1, using Python 3.6.<sup>10</sup>

The corresponding results can be seen in Table 6.4. It appears that our models exhibit adequate performance on identifying whether narration follows in the next time steps. Interestingly, for both the Timing@1 and the Timing@5 schemes, the multimodal variants outperforms the unimodal one, with our multi-view model scoring best for both setups.

### 6.3.3 Experiments on Content Generation

We experiment with several variations of the model described in Section 6.2 and some simpler, text-only retrieval baselines. Note that the models described in this section are trained and evaluated in dialogue/narration pairs; a different split than that used for narration timing. Moreover, for the experiments described in this section, we assume that timing information is given by an oracle.

---

<sup>9</sup><http://pytorch.org/>

<sup>10</sup><https://www.python.org/>

model	pr	rec	fi
T@1, text-only	52.8	71.9	60.9
T@1, multimodal	58.8	69.4	63.6
T@1, multimodal, MV	<b>62.5</b>	69.3	<b>65.8</b>
T@5, text-only	48.7	63.9	55.3
T@5, multimodal	55.3	61.0	58.0
T@5, multimodal, MV	<b>56.3</b>	61.1	<b>59.0</b>

Table 6.4: Results for narration timing, using the Timing@1 (T@1) and Timing@5 (T@5) annotations. Multimodal variants use information from image, audio and text modalities. MV stands for our multi-view model.

The retrieval baselines work as follows: for each dialogue of the test split, instead of generating a narration from scratch, they select one from the training set, by identifying the dialogue from the training set that is closer to it. All presented retrieval methods use *cosine similarity* as a similarity metric. Four retrieval baselines are tested:

- **Retrieval-avg** represents both narrations and dialogues by the mean of the word embeddings of their tokens.
- **Retrieval-tfidf** uses Term Frequency - Inverse Document Frequency (TF-IDF) representation for dialogues and narrations.
- **Retrieval-BERT** makes use of Bidirectional Encoder Representations from Transformers (BERT; Devlin et al. 2019) from a pre-trained model to represent both narrations and dialogues.<sup>11</sup>
- **Retrieval-CCA** uses Canonical Correlation Analysis (CCA; Hotelling 1935) to project the dialogue and narration spaces to a shared space. At test time, dialogue representations are projected to the shared space and the narrations whose projections are closest to them are selected. For this baseline, TF-IDF vectors are used for both dialogues and narrations. The CCA dimensionality is 300.<sup>12</sup>

We compare the output of the baselines with that of several variants of the *DiViNa* model:

- **DiViNa** is the model depicted on Figure 6.6(a), which makes use of text, audio and

<sup>11</sup>We used bert-as-a-service (<https://github.com/hanxiao/bert-as-service>) with the BERT-Base, Uncased pre-trained model.

<sup>12</sup>We also experimented with 100, 500 and 1000; all yielded lower performance.



	model	B-1	B-2	B-3	R-L	METEOR	CIDEr
TEXT	Retrieval-avg	14.59	7.45	4.38	13.60	6.89	21.94
	Retrieval-tfidf	17.73	10.37	6.18	16.43	9.16	30.84
	Retrieval-BERT	14.83	7.81	4.70	13.98	7.16	19.41
	Retrieval-CCA	13.45	7.45	4.74	12.77	6.39	21.20
MULTIMODAL	DiNa+att	8.19	3.88	2.34	12.00	4.84	4.36
	DiViNa	17.35	9.65	6.22	15.25	7.99	24.02
	DiViNa+att	15.70	8.59	5.82	13.71	6.88	25.42
	DiViNa, MV	21.22	13.60	10.22	18.08	9.52	48.40
	Di <sup>2</sup> ViNa	17.67	10.26	6.76	15.23	8.17	23.61
	Di <sup>2</sup> ViNa+att	18.45	9.88	5.94	15.54	7.82	22.04
	Di <sup>2</sup> ViNa, MV	19.37	10.85	6.95	16.70	8.08	16.70
	DiViNa+mmd	23.20	13.71	8.95	20.51	11.15	49.29
	DiViNa+att+mmd	20.43	12.03	7.86	17.71	9.22	40.82
	DiViNa+mmd, MV	24.15	14.39	9.30	21.73	<b>12.24</b>	56.35
	Di <sup>2</sup> ViNa+mmd	<b>24.57</b>	<b>15.90</b>	<b>11.72</b>	<b>21.70</b>	12.06	<b>67.24</b>
	Di <sup>2</sup> ViNa+att+mmd	22.93	14.79	10.36	19.96	10.74	52.93
	Di <sup>2</sup> ViNa+mmd, MV	23.78	14.94	10.82	21.29	11.06	64.86

Table 6.5: Results for narration content generation. BLEU-1 (B-1), BLEU-2 (B-2), BLEU-3 (B-3), ROUGE-L (R-L), METEOR and CIDEr metrics are reported. The first section of the table refers to baselines that use only textual information, while the rest of the table refers to multimodal models. The third section includes results for models that use the multimodal decoder (mmd), thus having information about the desired length of the narration. The models in the second section of the table do not have access to this information. The highest values are in boldface, while the best results for models that do not use the mmd are in italics.

image information from the dialogue before the narration and also encodes the video (image and audio information) of the narration part. The output of the two encoders is concatenated and passed through a linear layer to reduce its size, before it is fed to the decoder.

- **DiViNa+att** is the *DiViNa* model equipped with an attention mechanism over the states of the dialogue encoder. For this and other attentive variants, we make use of dot product attention (Luong et al., 2015), where the attention scores are calculated

as follows:

$$\text{score}(s_t, h_i) = s_t^\top h_i,$$

for each hidden state of encoder  $h_i$  and decoder hidden state  $s_t$ .

- **DiNa+att** is a variant of the *DiViNa+att* model, where there is no video encoder: the narration is generated based on the preceding dialogue only.
- **Di<sup>2</sup>ViNa, Di<sup>2</sup>ViNa+att**: Motivated by the omniscience of the narrator in the dataset, these models make use of the dialogue that follows the video to be narrated. This is achieved by encoding the dialogue that follows the narration using a separate dialogue encoder and feeding the future dialogue vector to the decoder. A schematic depiction of this model can be found in Figure 6.6(b).
- **DiViNa+mmd, Di<sup>2</sup>ViNa+mmd**: The decoder we have described so far makes use of audiovisual information only on its initialisation step. In order to be able to use the audio and image representations of the narration part, we experiment with a multi-modal decoder (mmd). The basic function of such a decoder is shown in Figure 6.7. During training, the fusion of image, audio and text representations is given to the decoder at each time step. Even when not using teacher forcing (i.e. when feeding the predictions of the previous step to the next step), the input of the decoder is the concatenation of the “correct” audiovisual representations and the word embedding of the token generated by the previous step. Since representations for image and audio are given in the dataset and not generated by the model, at test time, decoding proceeds for as many steps as the number of audiovisual representations available. We experiment with using the multimodal decoder on both *DiViNa* and *Di<sup>2</sup>ViNa*.
- **multi-view (MV) variants**: We also experiment with a multi-view variant of our model, that uses the correlational GRU, instead of an LSTM for the dialogue narration part.<sup>13</sup>

An obvious limitation of the multimodal decoder of the above model is that it cannot generate narrations of arbitrary length. This is not necessarily a constraint though, since some narration setups call for narrations of specific predefined length, as is the case for the Peppa Pig dataset. Since multimodal information is necessary, quantisation of the video chunk to be narrated should be done before the start of the decoding process. The known quantisation assumption is quite fair, since an estimation of the length of the narration will most

<sup>13</sup>We also experimented with using corrGRU cells in other encoders of our architecture, but using a multi-view model for the dialogue encoder yields the best performance.

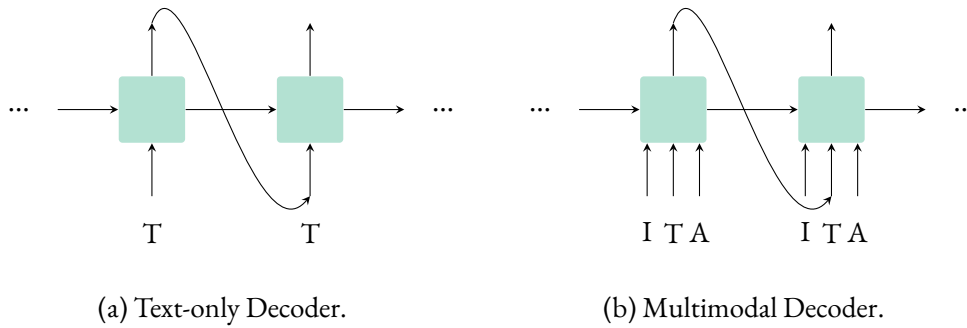


Figure 6.7: Multimodal Decoder used in *DiViNa+mmd* and *Di<sup>2</sup>ViNa+mmd* models and comparison with Text-only Decoder. At each timestep, audio (A) and image (I) representations are fed to the decoder, along with the word embedding of the corresponding token (T). In this particular case, teacher forcing is not used: the word embedding of the previously generated token is fused with the “correct” audiovisual vectors.

likely be available in all situations (for example, it can be calculated by dividing the time of the gap between two dialogues with the mean time it takes a narrator to utter a word). In our experiments, we exploited the known tokenisation of narrations to extract the multimodal representations needed to feed the decoder. As such, this model variant has access to more information than its counterparts.

In all models that contain a video encoder for the narration part, we made the simplifying choice to use the correct tokenisation of the narration, which is given in the dataset, despite the fact that, normally, a narrator model will not have access to that information.

**Implementation Details** For our experiments we used the following settings:

- All modalities are passed through a linear layer that yields embeddings of size 300.
- For the multi-view models, the 300 dimensional vectors are passed to the sequence model. For the early fusion model, the representations are concatenated; consequently, the input to the sequence model is of size 900.
- The hidden size of all the LSTMs, in the encoders and the decoder is 500.
- The output of more than one encoders is concatenated and passed through a bridge layer with output dimensionality of 500 before being fed to the decoder.
- We use one-layer sequence models for all experiments.

- Optimisation is done using the Adam optimiser, with an initial learning rate of 0.001.
- Training proceeds for 100 epochs.
- The model was implemented in PyTorch version 0.4.1, using Python 3.6.

Narration generation is a natural language generation (NLG) task, and as such, its evaluation is not straightforward. Automatic evaluation of NLG systems is an open problem and widely used metrics are under heavy criticism by the community (Callison-Burch et al., 2006; Liu et al., 2016). Following previous work on NLG, ranging from image captioning (Xu et al., 2015b) to summarization (See et al., 2017), we report a set of word overlap metrics, complementary to each other.<sup>14</sup> While these metrics have been used extensively in the context of image and video captioning, their nature allows for their use in text generation setups that do not necessarily have an image component. In the case of narration generation, they compare narrations from the dataset with narrations generated by our systems, without considering the video. The scores for all models can be seen in Table 6.5.

The generally low scores of the retrieval models suggest that the task of narration generation is not a trivial one and highlights the need for more sophisticated generative models. Quite interestingly though, the simple TF-IDF retrieval baseline not only outperforms the other text-only retrieval baselines, but also yields performance on par with that of some of the simplest generation multimodal models.

The significantly lower performance of *DiNa+att* suggests that encoding the video of the part to be narrated (that is what differentiates *DiNa* from *DiViNa*) is of high importance. Equally important is encoding the upcoming dialogue, as suggested by the generally higher scores of the *Di<sup>2</sup>ViNa* variants. The attentive variants of all models seem to perform lower than their non-attentive counterparts. A reason for that may be that they base their predictions more on the preceding dialogue (since they attend to it), while the contributions of the other encoders are downgraded. These results highlight a distinctive quality of the task of narration generation: the significance of effective combination of information from the past (preceding dialogue), the present (video to be narrated), and the future (upcoming dialogue).

Moreover, it can be seen that, in most setups, the multi-view variants of the narration generation models are able to outperform their non-multi-view counterparts. The fact that variants that use the multimodal decoder (mmd) outperform all other variants does not come as a surprise, since they have access to the audiovisual information at the time of decoding and also

<sup>14</sup>We used `nlg-eval`, available in <https://github.com/Maluuba/nlg-eval>.

generate narrations of the right length, by design. Some examples of generated narrations are shown in Figure 6.8.

<b>MODEL:</b> Peppa and jumping up and down in muddy puddles. Everyone loves jumping up.
<b>GROUND TRUTH:</b> Everyone in the whole world loves jumping up and down in muddy puddles.
<b>MODEL:</b> Suzy sheep has come to play on the.
<b>GROUND TRUTH:</b> Peppa has come to play with Suzy sheep.
<b>MODEL:</b> It is bedtime for Peppa and George are very sleepy.
<b>GROUND TRUTH:</b> It is nighttime. Peppa and George are going to bed.

Figure 6.8: Examples of the output of Di<sup>2</sup>ViNa+mmd model, paired with respective ground truth narrations.

## 6.4 Conclusions

In this chapter, we introduce and formalise the task of narration generation from videos. Narration generation refers to the task of accompanying videos with text snippets in several places; text that is meant to be uttered by a speaker and become part of the final video. The task of narration generation adds to the set of research tasks related to text generation from multimodal input. The problem includes a timing (figuring out when to narrate) and a content generation (figuring out what to include in the narration) part. Due to this dual nature of the problem, along with the fact that the content of narrations is, in general, context-aware and not particularly descriptive of the images shown in the video, we believe that this is a new and challenging task.

To facilitate research in the direction of automatic narration generation, we create a cartoon video dataset from the animated television series Peppa Pig, whose episodes include narration. We propose neural models to tackle both the problems of narration timing and narration content generation and report the first results on the newly introduced task and dataset.

# Chapter 7

## Conclusions

Multi-view learning algorithms are powerful representation learning tools, often exploited in the context of multimodal problems. This thesis discusses the applicability and performance of correlational multi-view representation learning techniques to a variety of problems related to NLP. We suggest that multi-view representation learning techniques should be seriously considered and used by researchers and practitioners of the NLP community. Especially for multimodal applications, research around multi-view representation learning can strengthen the relationship between communities (e.g. between the CV and NLP communities). The process of investigating common representations for multimodal data brings together ideas and researchers from the two communities, as opposed to the isolated approach of using CV or NLP modules as blackbox feature generators.

Although there is large amount of work on representation learning for multi-view data, commonly, related techniques are tested on synthetic datasets or strictly limited benchmarks. Arguably, such thorough examination of new techniques in well-researched and established datasets and setups is good practice, since it allows for clean and isolated evaluation. However, the full potential of new ideas is not realised unless they prove their ability to scale to the size, complexity and requirements of downstream tasks. We experiment with a wide range of applications, showing that representations created using such techniques can increase performance in various problems and setups.

Multi-view representations proved quite informative in the task of document summarisation, despite the unimodal nature of data. For abstract image captioning, using multi-view representations, we were able to generate captions of higher quality and diversity than those generated by non-multi-view setups. Moreover, a large part of the work of this thesis focuses

on video data; a type of dataset that is not yet fully explored by the Computer Vision and Natural Language Processing communities. We provide evidence on the benefits of adopting a multi-view framework while working with videos, as opposed to the early fusion paradigm. Shallow tasks, such as the structural, dialogue and plot tagging tasks that we examine using television series data, as well as text generation tasks, such as narration generation, benefit from a multi-view contemplation.

## 7.1 Future Work

There are several avenues for future research, based on the work presented in this thesis. The purpose of this section is to specify and comment on some of those directions.

### 7.1.1 Applications of Multi-view Representation Learning

In the task of document summarisation (see Chapter 3) we show that correlational multi-view representations are informative and yield adequate performance on the downstream task. However, the simple sentence selection setup we employ with those representations, encounters limitations in getting state-of-the-art performance on the CNN/DailyMail dataset.

Recent extractive and abstractive document summarisation systems use a range of mechanisms that are not present in our selection system: attention mechanisms that focus on different places of the document (Cheng and Lapata, 2016) or side information (Narayan et al., 2018a), copying mechanisms able to copy words straight from the document (out-of-vocabulary or otherwise; See et al. 2017), hierarchical attention (Nallapati et al., 2016b) and others. It is worth exploring then, whether our multi-view system and these state-of-the-art techniques can be combined. Further work on combining multi-view representations with representations from state-of-the-art systems can shed more light on the quality and effectiveness of the representations, revealing whether they are complementary.

In a similar fashion, given its performance, our CCA-based abstract scene captioning system described in Chapter 4, can be utilised to complement other image captioning systems. Image captioning systems operating on photographic data and yielding state-of-the-art performance can be complemented with representations inspired by our approach.

In Chapter 5, we present multi-view models for movie understanding tasks. While the output of tagging may not be too informative on its own, such tasks provide ground towards automatic movie understanding. Moreover, they can prove quite useful in information re-

trieval applications, where episodes or videos can be automatically enriched with a large set of deduced features or labels that facilitate search, without the need for explicit annotation, which can be costly and laborious. In the case of the CSI series, these labels can be tags indicating whether the episode has one or two crime cases, whether a large part of the episode is devoted to conversation between detectives and others.

In Chapter 6, we present the task of narration generation. Although we experiment with a simple cartoon dataset, our work is a step towards the direction of narrative content generation. Several problems pertaining to narrative content can be modelled by taking inspiration from our work, including generation of sports commentary, directors' commentaries in movies, or narrative content for news summary videos. The ideas, models and experiments presented in Chapter 6 are presented within a rather general and abstract framework which allows for easy extensions. For example, modelling narration timing as an incremental tagging problem was not strictly necessary for our specific dataset. Depending on the type of narration and the dataset, timing can be fairly straightforward (fill in gaps in the dialogue with length more than a couple of seconds; this is the case with our dataset), or could be more complex (decide when it is time for a narrator to speak, given the information they have at hand, as is the case in sports commentary). We chose to work with the more general case, providing a framework for benchmarking narrative content generation techniques.

### 7.1.2 Peppa Pig Dataset

In Chapter 6 we present a new video dataset from the animated cartoon series *Peppa Pig*. A strong point of the Peppa Pig dataset is the simplicity of its language and images. While it follows the paradigm of Abstract Scenes Dataset (Zitnick and Parikh 2013; also see Chapter 4) by abstracting away from complex images, at the same time it calls for an image processing component, as it does not use any semantic representations for images. As such, it stands in the middle between semantically annotated and complex, real-life datasets.

This dataset served as a testbed for the task of narration generation that is described in detail in Chapter 6. However, movie and cartoon datasets can have multiple uses. The Peppa Pig dataset can be used for episode creation, namely creating whole episodes by generating dialogues and narrations from the the video signal alone. Moreover, it can be used for video summarisation, video title generation and video description tasks.

The task of narration generation itself, as shown, needs inference on the plot level and is more creatively solved when information from the past, the present and the future is com-



bined. Further investigation using more complex models and/or architectures can yield better performance on this task. Finally, using other cartoon datasets, such as ToonNet ([Zhou et al., 2018](#)) for pre-training, can be beneficial for narration generation and other tasks on this dataset.

# Appendix A

## Implementation Details

### A.1 Encoder model used in Deep CCA Extractive Summariser (DCorES + encoder)

This appendix outlines the structural and implementation details of the encoder DCCA model, referenced in section 3.3.3.1. We repeat the schematic of the model in Figure A.1, for reference.

The model is heavily inspired by the architecture described in past work (Narayan et al., 2018a) and consists of a hierarchical encoder for the documents and a sentence encoder for the summary sentences. The encoded information is then fed to the DCCA model, and the whole network is trained by maximising the correlation between the two views.

More specifically,

- Data are tokenised and replaced with token ids. The vocabulary size is set to 150,000.
- Each sentence is encoded in a 350-dimensional vector by applying convolution. For the convolutional encoder, we followed (Kim, 2014), and used a list of kernels of widths 1 to 7, each with output channel size of 50.
- Sentences of each document are fed to an RNN network. For the recurrent neural network component in document encoder and sentence extractor, we used a single-layered LSTM network with size 600.
- Summary sentences are encoded in a 350-dimensional vector, by applying convolution, in a similar manner to the document’s sentences. The final hidden state of this

encoder is used as the first view input to the DCCA network.

- For every summary, we consider three sentences, using padding if the summary consists of less than three sentences, and selecting the first three in cases the summary consists of more than three sentences.
- The three vectors for every highlight are concatenated to give the second view input.
- The DCCA network lengths are set to [300, 800, 800, 300]. The output dimensionality is 300.
- The metric used for sentence similarity in decoding is cosine similarity.
- The model was trained with the Momentum optimiser with learning rate and momentum set to  $10^{-5}$ , and 0.99 respectively.
- The model is implemented in TensorFlow<sup>1</sup> (version 1.1.0) using Python 2.7.<sup>2</sup>

---

<sup>1</sup><https://www.tensorflow.org/>

<sup>2</sup><https://www.python.org/>

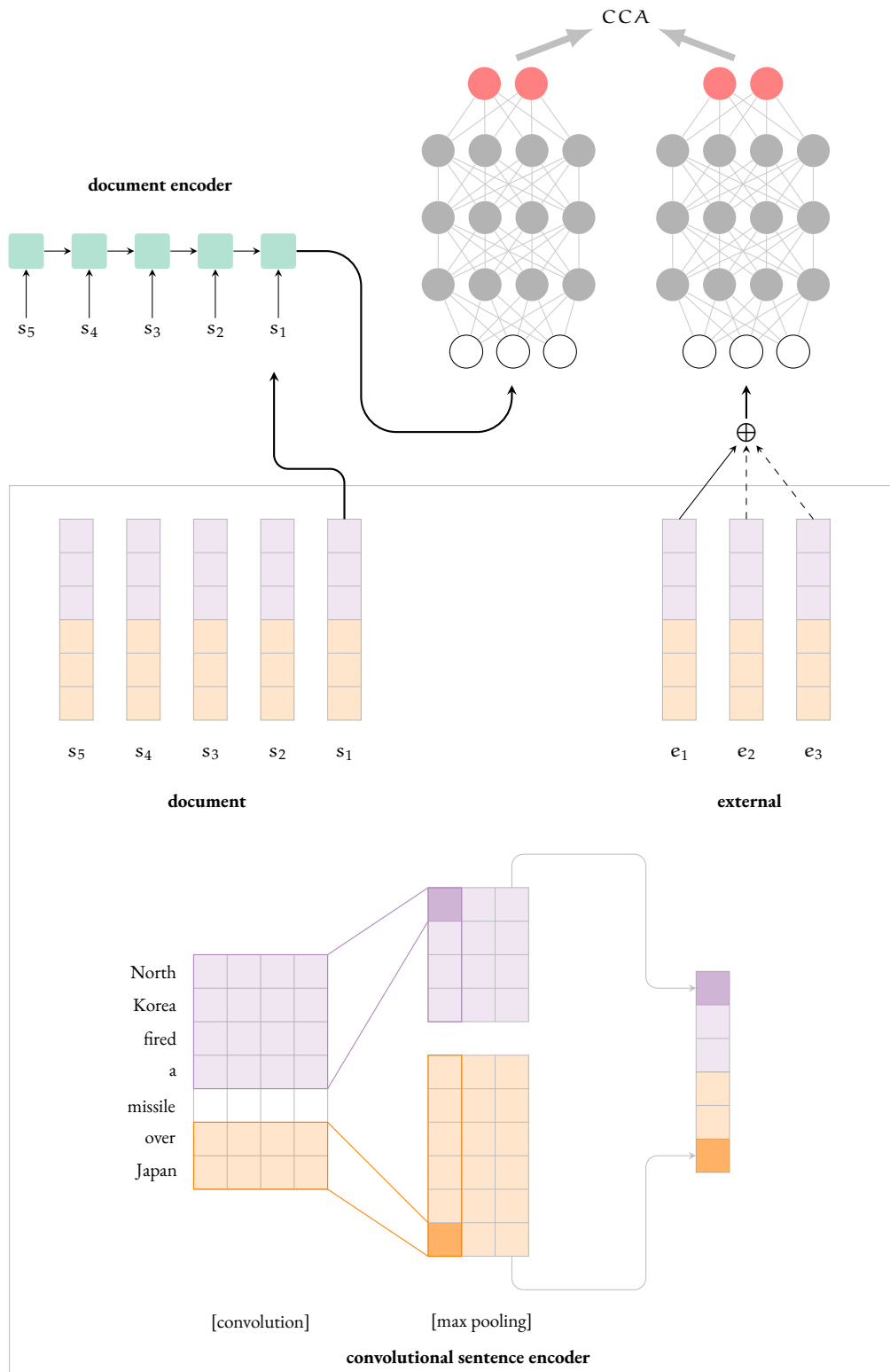


Figure A.1: Schematic of the encoder-dcca model.

## A.2 Multi-view sequential inference model

This appendix outlines the structural and implementation details of the multi-view inference model, referenced in Section 5.2.

- The hidden size of the LSTM and the corrGRU is 128.
- We use one-layer sequence models for all experiments.
- We set value for parameters  $\lambda = 0.001, \lambda_a = 0.001, \lambda_b = 0.001, \lambda_c = 0.001$  after tuning in the validation set.
- Optimisation is done using the Adam optimiser (Kingma and Ba, 2014), with an initial learning rate of 0.001.
- Training proceeds for 150 epochs.
- The model is implemented in TensorFlow<sup>3</sup> (version 1.7.0) using Python 2.7.<sup>4</sup>

---

<sup>3</sup><https://www.tensorflow.org/>

<sup>4</sup><https://www.python.org/>

# Appendix B

## Peppa Pig Dataset Details

### B.1 List of Episodes

This section contains a full list of the episodes used for the Peppa Pig Dataset described in Section 6.1.

ep. no	episode title	ep. no	episode title
season 1		season 4	
1	Muddy Puddles	106	Work and Play
2	Mr. Dinosaur is Lost	107	The Rainbow
3	Best Friend	108	Pedro's Cough
4	Polly Parrot	109	The Library
5	Hide and Seek	110	The Camper Van
6	The Playgroup	111	Camping Holiday
7	Mummy Pig at Work	112	Compost
8	Piggy in the Middle	113	Richard Rabbit Comes to Play
9	Daddy Loses his Glasses	114	Fun Run
10	Gardening	115	Washing
11	Hiccups	116	Polly's Boat Trip
12	Bicycles	117	Delphine Donkey
13	Secrets	118	The Fire Engine
14	Flying a Kite	119	Princess Peppa
15	Picnic	120	Teddy Playgroup
16	Musical Instruments	121	Danny's Pirate Party

17	Frogs and Worms and Butterflies	122	Mr. Potato Comes to Town
18	Dressing Up	123	The Train Ride
19	New Shoes	124	Granny Pig's Chickens
20	The School Fete	125	Talent Day
21	Mummy Pig's Birthday	126	A Trip To the Moon
22	The Tooth Fairy	127	Grandpa at the Playground
23	The New Car	128	Goldie the Fish
24	Treasure Hunt	129	Funfair
25	Not Very Well	130	Numbers
26	Snow	131	Digging up the Road
<b>season 2</b>		<b>season 6</b>	
27	Windy Castle	132	Freddy Fox
28	My Cousin Chloe	133	Whistling
29	Pancakes	134	Doctor Hamster's Tortoise
30	Babysitting	135	Sun, Sea and Snow
31	Ballet Lesson	136	Grandpa Pig's Computer
32	Thunderstorm	137	Hospital
33	Cleaning the Car	138	Spring
34	Lunch	139	Miss Rabbit's Helicopter
35	Camping	140	Baby Alexander
36	The Sleepy Princess	141	Grampy Rabbit's Lighthouse
37	The Tree House	142	Miss Rabbit's Day Off
38	Fancy Dress Party	143	The Secret Club
39	The Museum	144	Grampy Rabbit's Boatyard
40	Very Hot Day	145	Shake, Rattle and Bang
41	Chloe's Puppet Show	146	Champion Daddy Pig
42	Daddy Gets Fit	147	Chatterbox
43	Tidying Up	148	Mr. Fox's Van
44	The Playground	149	Chloe's Big Friends
45	Daddy Puts up a Picture	151	The Blackberry Bush
46	At the Beach	152	Pottery
47	Mister Skinnylegs	153	Paper Aeroplanes
48	Grandpa Pig's Boat	154	Edmond Elephant's Birthday
49	Shopping	155	The Biggest Muddy Puddle in the World
50	My Birthday Party	156	Santa's Grotto

51	Daddy's Movie Camera	157	Santa's Visit
52	School Play	<b>season 7</b>	
<b>season 3</b>		158	Potato City
53	Bubbles	159	The New House
54	Emily Elephant	160	Basketball
55	Polly's Holiday	161	Horsey Twinkle Toes
56	Teddy's Day Out	162	Naughty Tortoise
57	Mysteries	163	Mr. Fox's Shop
58	George's Friend	164	Shadows
59	Mr. Scarecrow	165	International Day
60	Windy Autumn Day	166	The Rainy Day Game
61	The Time Capsule	167	Mummy Rabbit's Bump
62	Rock Pools	168	Pedro the Cowboy
63	Recycling	169	Peppa and George's Garden
64	The Boat Pond	170	The Flying Vet
65	Traffic Jam	171	Kylie Kangaroo
66	Bedtime	172	Captain Daddy Dog
67	Sports Day	173	Grampy Rabbit's Dinosaur Park
68	The Eye Test	174	Bedtime Story
69	Granddad Dog's Garage	175	Lost Keys
70	Foggy Day	176	George's New Dinosaur
71	Jumble Sale	177	Grandpa Pig's Train to the Rescue
72	Swimming	178	The Pet Competition
73	Tiny Creatures	179	Spider Web
74	Daddy Pig's Office	180	The Noisy Night
75	Pirate Island	181	The Wishing Well
76	George Catches a Cold	182	Mr. Potato's Christmas Show
77	The Balloon Ride	183	Madame Gazelle's Leaving Party
78	George's Birthday	184	The Queen
<b>season 4</b>		<b>season 8</b>	
79	The Long Grass	185	Desert Island
80	Zoe Zebra the Postman's Daughter	186	Perfume
81	Painting	187	The Children's Fete
82	Cuckoo Clock	188	The Aquarium
83	The Baby Piggy	189	George's Racing Car



84	Grandpa's Little Train	190	The Little Boat
85	The Cycle Ride	191	The Sandpit
86	Ice Skating	192	Night Animals
87	The Dentist	193	Flying on Holiday
88	Dens	194	The Holiday House
89	Pretend Friend	195	Holiday in the Sun
90	School Bus Trip	196	The End of the Holiday
91	Rebecca Rabbit	197	Mirrors
92	Nature Trail	198	Pedro is Late
93	Pen Pal	199	Garden Games
94	Granny and Grandpa's Attic	200	Going Boating
95	The Quarrel	201	Mr. Bull in a China Shop
96	The Toy Cupboard	202	Fruit
97	School Camp	203	George's Balloon
98	Captain Daddy Pig	204	Peppa's Circus
99	The Powercut	205	The Fish Pond
100	Bouncy Ball	206	Snowy Mountain
101	Stars	207	Grampy Rabbit in Space
102	Daddy Pig's Birthday	208	The Olden Days
103	Sleepover	209	Pirate Treasure
104	Cold Winter Day		

## B.2 Preprocessing

This section outlines the preprocessing process followed for the episodes of the Peppa Pig Dataset, described in Section 6.1. Specifically, each episode was preprocessed as follows:

- Double episodes<sup>1</sup> were split to two separate episodes and corresponding subtitles were manually synchronised with the video.
- Each episode was trimmed to the correct length. Commonly, episodes start with an “introduction” part, in which Peppa, the main character, introduces herself and her family. This part, which usually lasts less than two minutes, was manually removed.
- The audio channel was extracted from each episode.
- We performed normalization of the text, by manually inspecting it. Since Peppa Pig is a cartoon targeting young children, animal sounds and sound effects are abundant in its dialogues. We manually identified and normalised such instances. As an example, all the occurrences of “oink”, “oink”, “oinking”, “big oink”, “loud snort” etc. were normalised to “\*pig\_sound\*” for text application and to a special noise token (“{ns}”) for forced alignment purposes (see next step).
- The subtitles were aligned with the audio channel in the token level. We used the Penn Forced Aligner (`p2fa`; Yuan and Liberman 2008) for that.
- After forced alignment, using `ffmpeg`,<sup>2</sup> we extracted frames (snapshots from the video) for every token, at timestamps corresponding to the beginning, the end and the middle of the token timeframe.
- Annotation of the narrator: the sentences uttered by the narrator were semi-manually annotated. Some subtitles, especially those that contained the first utterance of the narrator after a dialogue between the characters, begin with a tag (“[NARRATOR]” – subtitles are mostly meant for the hearing impaired, since Peppa Pig is aimed at preschoolers). However, the use of this cue is not consistent, and also, it is not used more than once in consecutive narrator subtitles. First, we annotated the subtitles beginning with the “NARRATOR” tag as belonging to the narrator, and a second manual pass was done to ensure that all narrations are properly tagged as such.
- MFCC features were calculated using the tool `aubio`.<sup>3</sup>

---

<sup>1</sup>most video files contained two episodes

<sup>2</sup><https://www.ffmpeg.org/>

<sup>3</sup><https://aubio.org/>



# Bibliography

- Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. 2016. Analyzing the behavior of visual question answering models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1955–1960.
- Rasim M Alguliev, Ramiz M Aliguliyev, Makrufa S Hajirahimova, and Chingiz A Mehdiyev. 2011. MCMR: Maximum coverage and minimum redundant text summarization model. *Expert Systems with Applications*, 38(12):14514–14522.
- Animashree Anandkumar, Rong Ge, Daniel J Hsu, Sham M Kakade, and Matus Telgarsky. 2014. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15(1):2773–2832.
- Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. SPICE: Semantic propositional image caption evaluation. In *European Conference on Computer Vision (ECCV)*, pages 382–398.
- Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(Nov):1817–1853.
- Galen Andrew, Raman Arora, Jeff A Bilmes, and Karen Livescu. 2013. Deep canonical correlation analysis. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1247–1255.
- Chee Siang Ang, Ania Bobrowicz, Diane J. Schiano, and Bonnie Nardi. 2013. Data in the wild: Some reflections. *Interactions*, 20(2):39–43.

- Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, Lawrence C Zitnick, and Devi Parikh. 2015. VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)*, pages 2425–2433.
- Relja Arandjelovic and Andrew Zisserman. 2017. Look, listen and learn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 609–617.
- Pradeep K Atrey, M Anwar Hossain, Abdulmotaleb El Saddik, and Mohan S Kankanhalli. 2010. Multimodal fusion for multimedia analysis: a survey. *Multimedia systems*, 16(6):345–379.
- Francis Bach and Michael Jordan. 2005. A probabilistic interpretation of canonical correlation analysis. Tech Report 688, Department of Statistics, University of California, Berkeley.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations (ICLR), San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2019. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):423–443.
- Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization@ACL 2005, Ann Arbor, Michigan, USA, June 29, 2005*, pages 65–72.
- Srinivas Bangalore, Vivek Kumar Rangarajan Sridhar, Prakash Kolan, Ladan Golipour, and Aura Jimenez. 2012. Real-time incremental speech-to-speech translation of dialogs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 437–445. Association for Computational Linguistics.
- Michele Banko, Vibhu O. Mittal, and Michael J. Witbrock. 2000. Headline generation based on statistical translation. In *38th Annual Meeting of the Association for Computational Linguistics (ACL), Hong Kong, China, October 1-8, 2000*.
- Leonard E. Baum and Ted Petrie. 1966. Statistical inference for probabilistic functions of finite state markov chains. *The Annals of Mathematical Statistics*, 37(6):1554–1563.

- Yoshua Bengio. 2008. Neural net language models. *Scholarpedia*, 3(1):3881.
- Yoshua Bengio, Patrice Simard, Paolo Frasconi, et al. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166.
- James Bennett, Stan Lanning, et al. 2007. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. New York, NY, USA.
- Adrian Benton, Raman Arora, and Mark Dredze. 2016. Learning multiview embeddings of twitter users. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 14–19.
- Adrian Benton, Huda Khayrallah, Biman Gujral, Dee Ann Reisinger, Sheng Zhang, and Raman Arora. 2019. Deep generalized canonical correlation analysis. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 1–6.
- Raffaella Bernardi, Ruket Cakici, Desmond Elliott, Aykut Erdem, Erkut Erdem, Nazli Ikizler-Cinbis, Frank Keller, Adrian Muscat, and Barbara Plank. 2016. Automatic description generation from images: A survey of models, datasets, and evaluation measures. *Journal of Artificial Intelligence Research*, 55:409–442.
- Guthrie S Birkhead, Michael Klompas, and Nirav R Shah. 2015. Uses of electronic health records for public health surveillance to advance public health. *Annual review of public health*, 36:345–359.
- Ali Furkan Biten, Lluís Gomez, Marçal Rusinol, and Dimosthenis Karatzas. 2019. Good news, everyone! context driven entity-aware captioning for news images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12466–12475.
- Ann Blair. 2012. [Information overload's 2,300-year-old history: Harvard business review online resources](http://blogs.hbr.org/cs/2011/03/information_overloads_2300-yea.html). [http://blogs.hbr.org/cs/2011/03/information\\_overloads\\_2300-yea.html](http://blogs.hbr.org/cs/2011/03/information_overloads_2300-yea.html), accessed 10 March, 2019.
- Avrim Blum and Tom M Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory (COLT), Madison, Wisconsin, USA, July 24-26, 1998.*, pages 92–100.
- Hervé Bredin, Anindya Roy, Nicolas Pécheux, and Alexandre Allauzen. 2014. "Sheldon speaking, bonjour!": Leveraging multilingual tracks for (weakly) supervised speaker identification. In *Proceedings of the 22nd ACM International Conference on Multimedia*.
- Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. 2017. Massive exploration

- of neural machine translation architectures. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1442–1451.
- Georgios-Ioannis Brokos, Prodromos Malakasiotis, and Ion Androutsopoulos. 2016. Using centroids of word embeddings and word mover’s distance for biomedical document retrieval in question answering. *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, pages 114 – 118.
- Ann L Brown, Joseph C Campione, and Jeanne D Day. 1981. Learning to learn: On training students to learn from texts. *Educational researcher*, 10(2):14–21.
- Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.
- Sergi Caelles, Jordi Pont-Tuset, Federico Perazzi, Alberto Montes, Kevis-Kokitsi Maninis, and Luc Van Gool. 2019. The 2019 DAVIS challenge on VOS: Unsupervised multi-object segmentation. *arXiv preprint arXiv:1905.00737*.
- Ozan Caglayan, Loïc Barrault, and Fethi Bougares. 2016. Multimodal attention for neural machine translation. *arXiv preprint arxiv:1609.03976*.
- Xiao Cai, Feiping Nie, and Heng Huang. 2013. Multi-view k-means clustering on big data. In *Twenty-Third International Joint conference on Artificial Intelligence (IJCAI)*.
- Iacer Calixto, Qun Liu, and Nick Campbell. 2017. Doubly-attentive decoder for multimodal neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers*, volume 1, pages 1913–1924.
- Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluation the role of BLEU in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.
- Ziqiang Cao, Chengyao Chen, Wenjie Li, Sujian Li, Furu Wei, and Ming Zhou. 2016. TG-Sum: Build tweet guided multi-document summarization dataset. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2906–2912.
- Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28:41–75.
- Sarath Chandar, Mitesh M. Khapra, Hugo Larochelle, and Balaraman Ravindran. 2016. Correlational neural networks. *Neural Computation*, 28:257–285.

- Xiaobin Chang, Tao Xiang, and Timothy M Hospedales. 2018. Scalable and effective deep CCA via soft decorrelation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1488–1497.
- David L Chen and William B Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 190–200. Association for Computational Linguistics.
- David L. Chen and Raymond J. Mooney. 2008. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the International Conference on Machine Learning (ICML)*.
- Minghai Chen, Sen Wang, Paul Pu Liang, Tadas Baltrušaitis, Amir Zadeh, and Louis-Philippe Morency. 2017. Multimodal sentiment analysis with word-level fusion and reinforcement learning. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pages 163–171.
- Ning Chen, Jun Zhu, and Eric P Xing. 2010. Predictive subspace learning for multi-view data: a large margin approach. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 361–369.
- Jiajun Cheng, Shenglin Zhao, Jiani Zhang, Irwin King, Xin Zhang, and Hui Wang. 2017. Aspect-level sentiment classification with HEAT (hierarchical attention) network. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 97–106. ACM.
- Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers*, volume 1, pages 484–494.
- Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *arXiv preprint arXiv:1606.02012*.
- Kyunghyun Cho, Bart Merriënboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.



- Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*.
- Cisco. 2019. Cisco visual networking index: Forecast and trends, 2017–2022. Technical report.
- Shay B Cohen, Karl Stratos, Michael Collins, Dean P Foster, and Lyle Ungar. 2012. Spectral learning of latent-variable PCFGs. In *The 50th Annual Meeting of the Association for Computational Linguistics, July 8-14, 2012, Jeju Island, Korea - Volume 1: Long Papers*, pages 223–231.
- Trevor Cohn and Mirella Lapata. 2008. Sentence compression beyond word deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING) - Volume 1*, pages 137–144.
- Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.
- Qiang Cui, Shu Wu, Qiang Liu, Wen Zhong, and Liang Wang. 2018a. MV-RNN: A multi-view recurrent neural network for sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering*.
- Yin Cui, Guandao Yang, Andreas Veit, Xun Huang, and Serge J. Belongie. 2018b. Learning to evaluate image captioning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5804–5812.
- Pradipto Das, Chenliang Xu, Richard F Doell, and Jason J Corso. 2013. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2634–2641.
- Sanjoy Dasgupta, Michael L Littman, and David A McAllester. 2002. PAC generalization bounds for co-training. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 375–382.
- Derek Dean and Caroline Webb. 2011. Recovering from information overload. *McKinsey Quarterly*, 1(1):80–88.

- Manfred Del Fabro and Laszlo Böszörményi. 2013. State-of-the-art and future challenges in video scene detection: a survey. *Multimedia systems*, 19(5):427–454.
- Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Volume 1 (Long and Short Papers)*, pages 4171–4186.
- Paramveer S Dhillon, Dean P Foster, and Lyle H Ungar. 2015. Eigenwords: Spectral word embeddings. *The Journal of Machine Learning Research*, 16(1):3035–3078.
- Bonnie Dorr, Christof Monz, Richard Schwartz, and David Zajic. 2005. A methodology for extrinsic evaluation of text summarization: does ROUGE correlate? In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 1–8.
- Timothy Dozat and Christopher D Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.
- Desmond Elliott and Frank Keller. 2013. Image description using visual dependency representations. In *Proceedings of the 2013 Conference in Empirical Methods in Natural Language Processing (EMNLP)*, volume 13, pages 1292–1302.
- Desmond Elliott and Frank Keller. 2014. Comparing automatic evaluation measures for image description. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL), June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*.
- Jeffrey L. Elman. 1990. Finding structure in time. *Cognitive Science*, 14:179–211.
- Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. 2018. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20:55:1–55:21.
- Günes Erkan and Dragomir R Radev. 2004. LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.
- Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian,

- Julia Hockenmaier, and David Forsyth. 2010. Every picture tells a story: Generating sentences from images. In *European Conference on Computer Vision (ECCV)*, pages 15–29.
- Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 462–471.
- Li Fei-Fei, Asha Muthuraman Iyer, Christof Koch, and Pietro Perona. 2007. What do we perceive in a glance of a real-world scene? *Journal of Vision*, 7(1):10.
- Minwei Feng, Bing Xiang, Michael R Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 813–820. IEEE.
- Elena Filatova and Vasileios Hatzivassiloglou. 2004. Event-based extractive summarization. In *Proceedings of ACL Workshop on Summarization*, volume III. Barcelona, Spain.
- Peter Flach. 2012. *Machine learning: the art and science of algorithms that make sense of data*. Cambridge University Press.
- David F Fouhey and C Lawrence Zitnick. 2014. Predicting object dynamics in scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2019–2026.
- Lea Frermann, Shay B Cohen, and Mirella Lapata. 2018. Whodunnit? crime drama as a case for natural language understanding. *Transactions of the Association of Computational Linguistics*, 6:1–15.
- Albert Gatt and Emiel Kraemer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.
- Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. 2017. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*.
- Felix A Gers and E Schmidhuber. 2001. LSTM recurrent networks learn simple context-free and context-sensitive languages. *IEEE Transactions on Neural Networks*, 12(6):1333–1340.
- Felix A Gers and Jürgen Schmidhuber. 2000. Recurrent nets that time and count. In *Pro-*

- ceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, volume 3, pages 189–194. IEEE.
- Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. <http://www.deeplearningbook.org>.
- Philip John Gorinski and Mirella Lapata. 2018. What’s this movie about? a joint neural network architecture for movie content analysis. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Volume 1 (Long Papers)*, volume 1, pages 1770–1781.
- Yvette Graham. 2015. Re-evaluating automatic summarization with BLEU and 192 shades of ROUGE. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP), Lisbon, Portugal, September 17-21, 2015*, pages 128–137.
- Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. 2016. LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232.
- Ralph Gross, Iain Matthews, Jeffrey Cohn, Takeo Kanade, and Simon Baker. 2008. Multi-pie. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*. IEEE Computer Society.
- Divya Gupta, Poonam Bansal, and Kavita Choudhary. 2018. The state of the art of feature extraction techniques in speech recognition. In *Speech and Language Processing for Human-Machine Communications*, pages 195–207. Springer.
- Pankaj Gupta, Yatin Chaudhary, and Hinrich Schütze. 2019. Multi-view and multi-source transfers in neural topic modeling with pretrained topic and word embeddings. *arXiv preprint arXiv:1909.06563*.
- Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL), June 15-20, 2008, Columbus, Ohio, USA*, volume 2008, pages 771–779.
- David R. Hardoon, Sándor Szedmák, and John Shawe-Taylor. 2004. Canonical correla-

- tion analysis: An overview with application to learning methods. *Neural Computation*, 16:2639–2664.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.
- Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. 2016. Generating visual explanations. In *European Conference on Computer Vision (ECCV)*.
- David Herman, Jahn Manfred, and Ryan Marie-Laure. 2010. *Routledge encyclopedia of narrative theory*. Routledge.
- Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1693–1701.
- Shawn Hershey, Sourish Chaudhuri, Daniel P. W. Ellis, Jort F. Gemmeke, Aren Jansen, R. Channing Moore, Manoj Plakal, Devin Platt, Rif A. Saurous, Bryan Seybold, Malcolm Slaney, Ron J. Weiss, and Kevin W. Wilson. 2017. CNN architectures for large-scale audio classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 131–135. IEEE.
- Margaret Hill. 1991. Writing summaries promotes thinking and learning across the curriculum: But why are they so difficult to write? *Journal of Reading*, 34(7):536–539.
- Geoffrey E Hinton. 1986. Learning distributed representations of concepts. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, volume 1, pages 1–12. Amherst, MA.
- Tsutomu Hirao, Masaaki Nishino, Jun Suzuki, and Masaaki Nagata. 2017. Enumeration of extractive oracle summaries. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (ACL): Volume 1, Long Papers*, pages 386–396.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.
- Micah Hodosh, Peter Young, and Julia Hockenmaier. 2013. Framing image description as

- a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899.
- Nils Holzenberger, Shruti Palaskar, Pranava Madhyastha, Florian Metze, and Raman Arora. 2019. Learning from multiview correlations in open-domain videos. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8628–8632. IEEE.
- Chiori Hori, Takaaki Hori, Teng-Yok Lee, Kazuhiro Sumi, John R. Hershey, and Tim K. Marks. 2017. Attention-based multimodal fusion for video description. In *IEEE International Conference on Computer Vision (ICCV)*, pages 4203–4212.
- Chiori Hori, Takaaki Hori, Gordon Wichern, Jue Wang, Teng-yok Lee, Anoop Cherian, and Tim K Marks. 2018. Multimodal attention for fusion of audio and spatiotemporal features for video description. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2528–2531.
- Paul Horst. 1961. Generalized canonical correlations and their applications to experimental data. *Journal of Clinical Psychology*, 17(4):331–347.
- Harold Hotelling. 1935. Canonical correlation analysis (CCA). *Journal of Educational Psychology*.
- Baotian Hu, Qingcai Chen, and Fangze Zhu. 2015. LCSTS: A large scale chinese short text summarization dataset. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1967–1972.
- Jagadeesh Jagarlamudi and Hal Daumé III. 2012. Regularized interlingual projections: evaluation on multilingual transliteration. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 12–23. Association for Computational Linguistics.
- Jesper Højvang Jensen, Mads Græsbøll Christensen, Manohar N Murthi, and Søren Holdt Jensen. 2006. Evaluation of MFCC estimation techniques for music similarity. In *14th European Signal Processing Conference*, pages 1–5. IEEE.
- Yangqing Jia, Mathieu Salzmann, and Trevor Darrell. 2010. Factorized latent spaces with structured sparsity. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 982–990.
- Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David A. Shamma, Michael S.

- Bernstein, and Li Fei-Fei. 2015. Image retrieval using scene graphs. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3668–3678.
- Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *International Conference on Machine Learning (ICML)*, pages 2342–2350.
- Samira Ebrahimi Kahou, Xavier Bouthillier, Pascal Lamblin, Çağlar Gülçehre, Vincent Michalski, Kishore Konda, Sébastien Jean, Pierre Froumenty, Yann N. Dauphin, Nicolas Boulanger-Lewandowski, Raul Chandias Ferrari, Mehdi Mirza, David Warde-Farley, Aaron C. Courville, Pascal Vincent, Roland Memisevic, Christopher Joseph Pal, and Yoshua Bengio. 2016. EmoNets: Multimodal deep learning approaches for emotion recognition in video. *Journal on Multimodal User Interfaces*, 10(2):99–111.
- Hidetaka Kamigaito, Katsuhiko Hayashi, Tsutomu Hirao, Hiroya Takamura, Manabu Okumura, and Masaaki Nagata. 2017. Supervised attention for sequence-to-sequence constituency parsing. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (IJCNLP), Volume 2: Short Papers*, volume 2, pages 7–12.
- Mert Kilickaya, Aykut Erdem, Nazli Ikizler-Cinbis, and Erkut Erdem. 2017. Re-evaluating automatic metrics for image captioning. In *15th Conference of the European Chapter of the Association for Computational Linguistics (EACL), Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*.
- Kyung-Min Kim, Min-Oh Heo, Seong-Ho Choi, and Byoung-Tak Zhang. 2017. DeepStory: video story QA by deep embedded memory networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2016–2022.
- KyungMin Kim, Min-Oh Heo, and Byoung-Tak Zhang. 2016. PororoQA: Cartoon video series dataset for story understanding. In *NIPS 2016 Workshop on Large Scale Computer Vision System*.
- Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference in Empirical Methods in Natural Language Processing (EMNLP)*. Citeseer.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. 2014. Multimodal neural language

- models. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 14, pages 595–603.
- Emily Kissner. 2006. *Summarizing, paraphrasing and retelling. Skills for Better Reading, Writing, and Test Taking*. Heinemann.
- Elena Knyazeva, Guillaume Wisniewski, Hervé Bredin, and François Yvon. 2015. Structured prediction for speaker identification in TV series. In *Sixteenth Annual Conference of the International Speech Communication Association*.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL), June 23-30, 2007, Prague, Czech Republic*, pages 177–180. Association for Computational Linguistics.
- Ben Krause, Iain Murray, Steve Renals, and Liang Lu. 2017. Multiplicative LSTM for sequence modelling. In *5th International Conference on Learning Representations (ICLR), Toulon, France, April 24-26, 2017, Workshop Track Proceedings*.
- Girish Kulkarni, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. 2011. Baby talk: Understanding and generating image descriptions. In *24th IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Colorado Springs, CO, USA, 20-25 June 2011*.
- Abhishek Kumar, Piyush Rai, and Hal Daumé III. 2011. Co-regularized multi-view spectral clustering. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1413–1421.
- Jayant Kumar, Qun Li, Survi Kyal, Edgar A Bernal, and Raja Bala. 2015. On-the-fly hand detection training with application in egocentric action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 18–27.
- John D. Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning (ICML)*.
- Chiraag Lala and Lucia Specia. 2018. Multimodal lexical translation. In *Proceedings of*



*the Eleventh International Conference on Language Resources and Evaluation (LREC), Miyazaki, Japan*, pages 3810–3817.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 260–270.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pages 2278–2324.

Greg Lee, Vadim Bulitko, and Elliot A Ludvig. 2014. Automated story selection for color commentary in sports. *IEEE transactions on Computational Intelligence and AI in Games*, 6(2):144–155.

Guang Li, Linchao Zhu, Ping Liu, and Yi Yang. 2019. Entangled transformer for image captioning. In *The IEEE International Conference on Computer Vision (ICCV)*.

Qing Li, Qingyi Tao, Shafiq Joty, Jianfei Cai, and Jiebo Luo. 2018. VQA-E: Explaining, elaborating, and enhancing your answers for visual questions. In *European Conference on Computer Vision (ECCV)*.

Paul Pu Liang, Ziyin Liu, AmirAli Bagher Zadeh, and Louis-Philippe Morency. 2018. Multimodal language analysis with recurrent multistage fusion. In *Proceedings of the 2018 Conference in Empirical Methods in Natural Language Processing (EMNLP)*.

Renars Liepins, Ulrich Germann, Guntis Barzdins, Alexandra Birch, Steve Renals, Susanne Weber, Peggy van der Kreeft, Herve Bourlard, João Prieto, Ondrej Klejch, Peter Bell, Alexandros Lazaridis, Alfonso Mendes, Sebastian Riedel, Mariana S. C. Almeida, Pedro Balage, Shay B. Cohen, Tomasz Dwojak, Philip N. Garner, Andreas Giefer, Marcin Junczys-Dowmunt, Hina Imran, David Nogueira, Ahmed Ali, Sebastião Miranda, Andrei Popescu-Belis, Lesly Miculicich Werlen, Nikos Papasrantopoulos, Abiola Obamuyide, Clive Jones, Fahim Dalvi, Andreas Vlachos, Yang Wang, Sibio Tong, Rico Senrich, Nikolaos Pappas, Shashi Narayan, Marco Damonte, Nadir Durrani, Sameer Khurana, Ahmed Abdelali, Hassan Sajjad, Stephan Vogel, David Sheppey, Chris Hernon, and Jeff Mitchell. 2017. The SUMMA platform prototype. In *Proceedings of the Software*

- Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 116–119, Valencia, Spain. Association for Computational Linguistics.
- Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Workshop on Text Summarization Branches Out, Post Conference Workshop of ACL 2004*.
- Chin-Yew Lin and FJ Och. 2004. Looking for a few good metrics: ROUGE and its evaluation. In *4th NTCIR Workshop*.
- Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*.
- Cailiang Liu, Dong Wang, Jun Zhu, and Bo Zhang. 2013. Learning a contextual multi-thread model for movie/TV scene segmentation. *IEEE Transactions on Multimedia*, 15:884–897.
- Chenxi Liu, Junhua Mao, Fei Sha, and Alan L Yuille. 2017a. Attention correctness in neural image captioning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 4176–4182.
- Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2122–2132.
- Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017b. Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1789–1798.
- Weifeng Liu, Dacheng Tao, Jun Cheng, and Yuanyan Tang. 2014. Multiview Hessian discriminative sparse coding for image annotation. *Computer Vision and Image Understanding*, 118:50 – 60.
- David Lopez-Paz, Suvrit Sra, Alex Smola, Zoubin Ghahramani, and Bernhard Schölkopf.

2014. Randomized nonlinear component analysis. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1359–1367.
- Ang Lu, Weiran Wang, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Deep multi-lingual correlation for improved word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.
- Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. 2017. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2018. Neural baby talk. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7219–7228.
- Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.
- Rebecca Mason and Eugene Charniak. 2014. Nonparametric method for data-driven image captioning. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL), June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, pages 592–598.
- Iain A. Matthews, Timothy F. Cootes, J. Andrew Bangham, Stephen J. Cox, and Richard Harvey. 2002. Extraction of visual features for lipreading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:198–213.
- Harry McGurk and John MacDonald. 1976. Hearing lips and seeing voices. *Nature*, 264(5588):746.
- Larry Medsker and Lakhmi C Jain. 1999. *Recurrent Neural Networks: Design and Applications*. CRC Press.
- Paul Mermelstein. 1976. Distance measures for speech recognition, psychological and instrumental. *Pattern recognition and artificial intelligence*, 116:374–388.
- Haitao Mi, Zhiguo Wang, and Abe Ittycheriah. 2016. Supervised attentions for neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2283–2288.

- Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.
- Xavier Anguera Miró, Simon Bozonnet, Nicholas W. D. Evans, Corinne Fredouille, Gerald Friedland, and Oriol Vinyals. 2012. Speaker diarization: A review of recent research. *IEEE Transactions on Audio, Speech, and Language Processing*, 20:356–370.
- Amy Mitchell, Elisa Shearer, Jeffrey Gottfried, and Michael Barthel. 2016. [The modern news consumer: News attitudes and practices in the digital era](http://www.journalism.org/2016/07/07/pathways-to-news/). <http://www.journalism.org/2016/07/07/pathways-to-news/>, accessed 10 March, 2019.
- Margaret Mitchell, Xufeng Han, Jesse Dodge, Alyssa Mensch, Amit Goyal, Alex Berg, Kota Yamaguchi, Tamara Berg, Karl Stratos, and Hal Daumé III. 2012. Midge: Generating image descriptions from computer vision detections. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 747–756. Association for Computational Linguistics.
- Mathew Monfort, Bolei Zhou, Sarah Adel Bargal, Alex Andonian, Tom Yan, Kandan Ramakrishnan, Lisa M. Brown, Quanfu Fan, Dan Gutfreund, Carl Vondrick, and Aude Oliva. 2019. Moments in time dataset: one million videos for event understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Louis-Philippe Morency, Rada Mihalcea, and Payal Doshi. 2011. Towards multimodal sentiment analysis: Harvesting opinions from the web. In *Proceedings of the 13th International Conference on Multimodal Interfaces*, pages 169–176. ACM.
- Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. 2016. Generating natural questions about an image. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1802–1813.
- Ramesh Nallapati, Bowen Zhou, and Mingbo Ma. 2016a. Classify or select: Neural architectures for extractive document summarization. *arXiv preprint arXiv:1611.04244*.
- Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çağlar Gülçehre, and Bing Xiang. 2016b. Abstractive text summarization using sequence-to-sequence RNNs and

- beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, Berlin, Germany, August 11-12, 2016, pages 280–290.
- Shashi Narayan, Ronald Cardenas, Nikos Papasarantopoulos, Shay B. Cohen, Mirella Lapata, Jiangsheng Yu, and Yi Chang. 2018a. Document modeling with external attention for sentence extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers*, pages 2020–2030.
- Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018b. Don't give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1797–1807. Association for Computational Linguistics.
- Ani Nenkova, Lucy Vanderwende, and Kathleen McKeown. 2006. A compositional context sensitive multi-document summarizer: exploring the factors that influence summarization. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 573–580. ACM.
- Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. 2011. Multimodal deep learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 689–696.
- Mihalis A Nicolaou, Hatice Gunes, and Maja Pantic. 2011. Continuous prediction of spontaneous affect from multiple cues and modalities in valence-arousal space. *IEEE Transactions on Affective Computing*, 2(2):92–105.
- Vicente Ordonez, Girish Kulkarni, and Tamara L Berg. 2011. Im2Text: Describing images using 1 million captioned photographs. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1143–1151.
- Luis Gilberto Mateos Ortiz, Clemens Wolff, and Mirella Lapata. 2015. Learning to interpret and describe abstract scenes. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, volume 3, pages 1505–1515.
- Dominique Osborne, Shashi Narayan, and Shay B. Cohen. 2016. Encoding prior knowledge with eigenword embeddings. In *Transactions of the Association for Computational Linguistics*.

- Miles Osborne. 2002. Using maximum entropy for sentence extraction. In *Proceedings of the ACL-02 Workshop on Automatic Summarization-Volume 4*, pages 1–8. Association for Computational Linguistics.
- Nikos Papasarantopoulos, Lea Frermann, Mirella Lapata, and Shay B. Cohen. 2019. Partners in crime: Multi-view sequential inference for movie understanding. In *2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.
- Nikos Papasarantopoulos, Helen Jiang, and Shay B. Cohen. 2018. Canonical correlation inference for mapping abstract scenes to text. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI), the 30th Innovative Applications of Artificial Intelligence (IAAI), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI), New Orleans, Louisiana, USA, February 2-7, 2018*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), July 6-12, 2002, Philadelphia, PA, USA*.
- Eric K. Patterson, Sabri Gurbuz, Zekeriya Tufekci, and John N. Gowdy. 2002. CUAVE: A new audio-visual database for multimodal human-computer interface research. *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2:II-2017–II-2020.
- Karl Pearson. 1901. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.
- Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 2227–2237.
- Steven Pinker. 2015. *The sense of style: The thinking person's guide to writing in the 21st century*. Penguin Books.

- Novi Quadrianto and Christoph Lampert. 2011. Learning multi-view neighborhood preserving projections. In *Proceedings of the 28th International Conference on Machine Learning (ICML), Washington, USA; 28 June-2 July 2011*, pages 425–432. Association for Computing Machinery.
- Stephan Raaijmakers, Khiet Truong, and Theresa Wilson. 2008. Multimodal subjectivity analysis of multiparty conversation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 466–474. Association for Computational Linguistics.
- Dragomir Radev, Timothy Allison, Sasha Blair-Goldensohn, John Blitzer, Arda Çelebi, Stanko Dimitrov, Elliott Drabek, Ali Hakim, Wai Lam, Danyu Liu, Jahna Otterbacher, Hong Qi, Horacio Saggion, Simone Teufel, Michael Topper, Adam Winkel, and Zhu Zhang. 2004. MEAD - a platform for multidocument multilingual text summarization. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC)*.
- Shyam Sundar Rajagopalan, Louis-Philippe Morency, Tadas Baltrušaitis, and Roland Goecke. 2016. Extending long short-term memory for multi-view structured learning. In *European Conference on Computer Vision (ECCV)*, pages 338–353. Springer.
- Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural Language Processing using Very Large Corpora*, pages 157–176. Springer.
- Pushpendre Rastogi, Ryan Cotterell, and Jason Eisner. 2016. Weighting finite-state transductions with neural context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 623–633.
- Pushpendre Rastogi, Benjamin Van Durme, and Raman Arora. 2015. Multiview LSA: Representation learning via generalized CCA. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 556–566.
- Alec Ratford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, Tech. Rep., Technical report, OpenAI.
- Lev-Arie Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named

- entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL)*, Boulder, Colorado, USA, June 4-5, 2009, pages 147–155.
- Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. 2013. Grounding action descriptions in videos. *Transactions of the Association of Computational Linguistics*, 1:25–36.
- James M. Rehg, Gregory D. Abowd, Agata Rozga, Mario Romero, Mark A. Clements, Stan Sclaroff, Irfan A. Essa, Opal Y. Ousley, Yin Li, Chanh Kim, Hrishikesh Rao, Jonathan C. Kim, Liliana Lo Presti, Jianming Zhang, Denis Lantsman, Jonathan Bidwell, and Zhefan Ye. 2013. Decoding children’s social behavior. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3414–3421. IEEE.
- Ehud Reiter and Anja Belz. 2009. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics*, 35:529–558.
- Jimmy S. J. Ren, Yongtao Hu, Yu-Wing Tai, Chuan Wang, Li Xu, Wenxiu Sun, and Qiong Yan. 2016. Look, listen and learn - a multimodal LSTM for speaker identification. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*.
- Peter Gordon Roetzel. 2018. Information overload in the information age: a review of the literature from business administration, business psychology, and related disciplines with a bibliometric approach and framework development. *Business Research*, pages 1–44.
- Anna Rohrbach, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Manfred Pinkal, and Bernt Schiele. 2014. Coherent multi-sentence video description with variable level of detail. In *German Conference on Pattern Recognition*, pages 184–195. Springer.
- Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. 2015. A dataset for movie description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3202–3212.
- Anna Rohrbach, Atousa Torabi, Marcus Rohrbach, Niket Tandon, Christopher Joseph Pal, Hugo Larochelle, Aaron C. Courville, and Bernt Schiele. 2016. Movie description. *International Journal of Computer Vision*, 123:94–120.
- Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. 2012. A



- database for fine grained activity detection of cooking activities. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1194–1201. IEEE.
- Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326.
- Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 379–389.
- Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.
- Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers*, volume 1, pages 1073–1083.
- Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers*, pages 2556–2565.
- Yue Shi, Martha Larson, and Alan Hanjalic. 2013. Mining contextual movie similarity with matrix factorization for context-aware recommendation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(1):16.
- Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations (ICLR), San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Gabriel Skantze. 2017. Towards a general, continuous model of turn-taking in spoken dialogue using LSTM recurrent neural networks. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 220–230.
- Michael A Smith and Takeo Kanade. 1998. Video skimming and characterization through the combination of image and language understanding. In *International Workshop on Content-Based Access of Image and Video Databases (CAIVD), Bombay, India, January 3, 1998*, pages 61–70. IEEE.
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, An-

- drew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1631–1642, Stroudsburg, PA. Association for Computational Linguistics.
- Swapna Somasundaran, Janyce Wiebe, Paul Hoffmann, and Diane Litman. 2006. Manual annotation of opinion categories in meetings. In *Proceedings of the Workshop on Frontiers in Linguistically Annotated Corpora 2006*, pages 54–61. Association for Computational Linguistics.
- Nitish Srivastava and Ruslan R Salakhutdinov. 2012. Multimodal learning with deep Boltzmann machines. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2222–2230.
- Karl Stratos, Michael Collins, and Daniel Hsu. 2016. Unsupervised Part-of-Speech tagging with anchor Hidden Markov Models. *Transactions of the Association for Computational Linguistics*.
- Karl Stratos, Do-kyum Kim, Michael Collins, and Daniel Hsu. 2014. A spectral algorithm for learning class-based n-gram models of natural language. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence (UAI), Quebec City, Quebec, Canada, July 23-27, 2014*, pages 762–771.
- Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5027–5038.
- Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G. Learned-Miller. 2015. Multi-view convolutional neural networks for 3D shape recognition. In *International Conference on Computer Vision (ICCV)*.
- Krysta Marie Svore, Lucy Vanderwende, and Christopher JC Burges. 2007. Enhancing single-document summarization by combining RankNet and third-party sources. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 448–457.
- Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, Inception-ResNet and the impact of residual connections on learning.

- In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, volume 4, page 12.
- Richard Szeliski. 2011. *Computer vision algorithms and applications*. Texts in computer science. Springer, London ; New York.
- Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2016. MovieQA: Understanding stories in movies through question-answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4631–4640.
- Y-I Tian, Takeo Kanade, and Jeffrey F Cohn. 2001. Recognizing action units for facial expression analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 23(2):97–115.
- Atousa Torabi, Christopher Pal, Hugo Larochelle, and Aaron Courville. 2015. Using descriptive video services to create a large data source for video annotation research. *arXiv preprint arXiv:1503.01070*.
- Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP, Rep4NLP@ACL 2017*, pages 191–200.
- Yao-Hung Hubert Tsai, Paul Pu Liang, Amir Zadeh, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Learning factorized multimodal representations. In *7th International Conference on Learning Representations (ICLR), New Orleans, LA, USA, May 6-9, 2019*.
- Grigorios Tzortzis and Aristidis Likas. 2012. Kernel-based weighted multi-view clustering. In *2012 IEEE 12th international conference on data mining*, pages 675–684. IEEE.
- Raghavendra Udupa and Mitesh M Khapra. 2010. Transliteration equivalence using Canonical Correlation Analysis. In *European Conference on Information Retrieval*, pages 75–86. Springer.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008.

- Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. CIDEr: Consensus-based image description evaluation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4566–4575.
- Alexei Vinokourov, Nello Cristianini, and John S Shawe-Taylor. 2002. Inferring a semantic representation of text via cross-language correlation analysis. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 1473–1480.
- Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164.
- Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. 2007. Manifold-ranking based topic-focused multi-document summarization. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, volume 7, pages 2903–2908.
- Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. 2016. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision (ECCV)*.
- Weiran Wang, Raman Arora, Karen Livescu, and Jeff Bilmes. 2015a. On deep multi-view representation learning. In *International Conference on Machine Learning (ICML)*, pages 1083–1092.
- Weiran Wang, Raman Arora, Karen Livescu, and Jeff A Bilmes. 2015b. Unsupervised learning of acoustic features via deep canonical correlation analysis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4590–4594. IEEE.
- Weiran Wang, Raman Arora, Karen Livescu, and Nathan Srebro. 2015c. Stochastic optimization for deep CCA via nonlinear orthogonal iterations. In *53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 688–695.
- Xiaogang Wang and Xiaoou Tang. 2008. Face photo-sketch synthesis and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):1955–1967.
- Gail Weiss, Yoav Goldberg, and Eran Yahav. 2018. On the practical computational power of finite precision RNNs for language recognition. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 2: Short Papers*, pages 740–745.

- John R. Westbury. 1994. Microbeam speech production database user’s handbook. In *IEEE Personal Communications*.
- Christopher KI Williams and Matthias Seeger. 2001. Using the Nyström method to speed up kernel machines. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 682–688.
- Ronald J Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280.
- Kristian Woodsend, Yansong Feng, and Mirella Lapata. 2010. Title generation with quasi-synchronous grammar. In *Proceedings of the 2010 Conference in Empirical Methods in Natural Language Processing (EMNLP)*.
- Qi Wu, Chunhua Shen, Peng Wang, Anthony Dick, and Anton van den Hengel. 2017a. Image captioning and visual question answering based on attributes and external knowledge. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1367–1381.
- Qi Wu, Damien Teney, Peng Wang, Chunhua Shen, Anthony Dick, and Anton van den Hengel. 2017b. Visual question answering: A survey of methods and datasets. *Computer Vision and Image Understanding*, 163:21–40.
- Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. 2015. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37:1834–1848.
- Cai Xu, Ziyu Guan, Wei Zhao, Yunfei Niu, Quan Wang, and Zhiheng Wang. 2018. Deep multi-view concept learning. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 2898–2904.
- Chang Xu, Dacheng Tao, and Chao Xu. 2013. A survey on multi-view learning. *arXiv preprint arxiv:1304.5634*.
- Chang Xu, Dacheng Tao, and Chao Xu. 2015a. Multi-view learning with incomplete views. *IEEE Transactions on Image Processing*, 24(12):5812–5825.
- Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwang Zhang, Xiangnan He, and Yueting Zhuang. 2017. Video question answering via gradually refined attention over appearance and motion. In *Proceedings of the 2017 ACM on Multimedia Conference*, pages 1645–1653. ACM.
- Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015b. Show, attend and tell: Neural image cap-

- tion generation with visual attention. In *International Conference on Machine Learning (ICML)*, pages 2048–2057.
- Fei Yan and Krystian Mikolajczyk. 2015. Deep correlation for matching images and text. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3441–3450.
- Xitong Yang, Palghat Ramesh, Radha Chitta, Sriganesh Madhvanath, Edgar A. Bernal, and Jiebo Luo. 2017. Deep multimodal representation learning from temporal data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5066–5074.
- Xiwang Yang, Harald Steck, and Yong Liu. 2012. Circle-based recommendation in online social networks. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1267–1275. ACM.
- Yezhou Yang, Ching Lik Teo, Hal Daumé III, and Yiannis Aloimonos. 2011. Corpus-guided sentence generation of natural images. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 444–454. Association for Computational Linguistics.
- Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1480–1489.
- Mahsa Yarmohammadi, Vivek Kumar Rangarajan Sridhar, Srinivas Bangalore, and Baskaran Sankaran. 2013. Incremental segmentation and decoding strategies for simultaneous translation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing (IJCNLP)*, pages 1032–1036.
- Wenpeng Yin and Yulong Pei. 2015. Optimizing sentence modeling and selection for document summarization. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1383–1389.
- Jun Yu, Jing Li, Zhou Yu, and Qingming Huang. 2019. Multimodal transformer with multi-view visual representation for image captioning. *IEEE Transactions on Circuits and Systems for Video Technology*.
- Youngjae Yu, Jongwook Choi, Yeonhwa Kim, Kyung Yoo, Sang-Hun Lee, and Gunhee Kim. 2017. Supervising neural attention models for video captioning by human gaze

- data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, Hawaii*, pages 2680–29.
- Jiahong Yuan and Mark Liberman. 2008. Speaker identification on the SCOTUS corpus. *Journal of the Acoustical Society of America*, 123(5):3878.
- Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2017. Tensor fusion network for multimodal sentiment analysis. In *Proceedings of the 2017 Conference in Empirical Methods in Natural Language Processing (EMNLP)*.
- Amir Zadeh, Paul Pu Liang, Navonil Mazumder, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018a. Memory fusion network for multi-view sequential learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5634–5641.
- Amir Zadeh, Paul Pu Liang, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018b. Multimodal language analysis in the wild: CMU-MOSEI dataset and interpretable dynamic fusion graph. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2236–2246.
- Amir Zadeh, Paul Pu Liang, Soujanya Poria, Prateek Vij, Erik Cambria, and Louis-Philippe Morency. 2018c. Multi-attention recurrent network for human communication comprehension. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI), the 30th Innovative Applications of Artificial Intelligence (IAAI), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5642–5649.
- Kuo-Hao Zeng, Tseng-Hung Chen, Ching-Yao Chuang, Yuan-Hong Liao, Juan Carlos Niebles, and Min Sun. 2017. Leveraging video descriptions to learn video question answering. In *Thirty-First AAAI Conference on Artificial Intelligence*, pages 4334–4340.
- Liang Zheng, Yi Yang, and Alexander G. Hauptmann. 2016. Person re-identification: Past, present and future. *arXiv preprint arXiv:1610.02984*.
- Wenming Zheng, Xiaoyan Zhou, Cairong Zou, and Li Zhao. 2006. Facial expression recognition using kernel Canonical Correlation Analysis (KCCA). *IEEE Transactions on Neural Networks*, 17(1):233–238.

- Yanqing Zhou, Yongxu Jin, Anqi Luo, Szeyu Chan, Xiangyun Xiao, and Xubo Yang. 2018. ToonNet: a cartoon image dataset and a DNN-based semantic classification system. In *Proceedings of the 16th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry*, page 30. ACM.
- Xinxin Zhu, Lixiang Li, Jing Liu, Haipeng Peng, and Xinxin Niu. 2018. Captioning transformer with stacked attention modules. *Applied Sciences*, 8(5):739.
- C Lawrence Zitnick and Devi Parikh. 2013. Bringing semantics into focus using visual abstraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3009–3016.
- C Lawrence Zitnick, Devi Parikh, and Lucy Vanderwende. 2013. Learning the visual interpretation of sentences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1681–1688.