ARTICLE SEARCH TOOL AND TOPIC CLASSIFIER

A Thesis

by

SANJEEV NARAYANAN

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Chao Tian
Co-Chair of Committee,	Anxiao Jiang
Committee Members,	Tie Liu
	Byung-Jun Yoon
Head of Department,	Miroslav Begovic

December 2019

Major Subject: Electrical Engineering

ABSTRACT

This thesis focuses on 3 main tasks related to Document Recommendations. The first approach deals with applying existing techniques on Document Recommendations using Doc2Vec. A robust representation of the same is presented to understand how noise induced in the embedding space affects predictions of the recommendations.

The next phase focuses on improving the above recommendations using a Topic Classifier. A Hierarchical Attention Network is employed for this purpose. In order to increase the accuracy of prediction, this work establishes a relation to embedding size of the words in the article.

In the last phase, model-agnostic Explainable AI (XAI) techniques are implemented to prove the findings in this thesis. XAI techniques are also employed to show how we can fine tune model hyper-parameters for a black-box model.

DEDICATION

To my parents and my dearest friends.

ACKNOWLEDGMENTS

I use this opportunity to express my gratitude to everyone who supported me throughout this extremely rewarding research experience. I extend my sincere appreciation to my advisor, Dr. Anxiao Jiang for guiding me in every step of this research work.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Professor Chao Tian [advisor], Professor Anxiao Jiang [co-advisor] and Professor Tie Liu & Professor Byung-Jun Yoon of the Department of Electrical and Computer Engineering and

All work conducted for the thesis dissertation was completed by the student independently.

Funding Sources

This thesis work was not funded by any source.

NOMENCLATURE

BiGRU	Bidirectional Gated Recurrent Unit
AI	Artificial Intelligence
XAI	Hierarchical Attention Network
PDP	Partial Dependence Plot
ICE	Individual Conditional Expectation
LIME	Locally Interpretable Model-agnostic Explanations
SHAP	Shapley Additive Explanations
ELI5	Explain Like I'm 5
Doc2Vec	Document-to-Vector
Word2Vec	Word-to-Vector
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
MTC	Medical Text Classification
NLP	Natural Language Processing
GloVe	Global Vectors
LSTM	Long-Short Term Memory
CBOW	Continuous Bag-of-Words

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION TO TOPICS

The human brain is wired to understand complex sentences and have the ability to draw multiple conclusions from a given statement. Over an individual's learning period, on an average, a person is exposed to roughly 20,000 to 40,000 different words in English. Along with the knowledge of grammar, the number of sentences one can comprehend is unbounded. We are in an era where we focus on automating human tasks to simplify effort. The discipline of Natural Language Processing (NLP) aims to improve the cognitive understanding of texts supplied to the machine. For example, consider a situation where you have read the story of "Snow White and the Seven Dwarves". It so happens that you forgot the names of characters, but you remember the main picture in the story. In order to find the name of the story to fit your flow of thought, you will instinctively Google for all possible phrases that would be coherent with the plot of the story. You may search for the combinations that are not limited to "young girl", "evil queen", "short people", "prince". Your required answer may not even be at the top of your search results. How exactly do we make this more accurate? How will we be able to narrow down our search to the exact answer or in other words, how will a machine be able to provide the result you are expecting? This happens to be a widely researched problem in NLP, which is the crux of my thesis.

Model interpretation at heart, is to find out ways to understand model decision making policies better. This is to enable accountability and transparency which will give users enough credence to use these models in real-world problems which have a lot of impact to business and society. Although, the black-box nature of these systems allow strong predictions, it cannot be precisely explained. AI has already become ubiquitous and we have become reliant on AI making decisions for us in our daily life, from merchandise and movie suggestions on Amazon and Netflix to friend recommendations on Facebook and personalized ads on Google search result pages. Nonetheless, in life-changing inferencing such as disease diagnosis, it is important to know the reasons behind such a critical decision. To address such issues, we use Explainable Artificial Intelligence(XAI) to

make a transition towards more transparent AI. It aims to create a suite of strategies that produce more explainable models whilst maintaining high performance levels.

2. ARTICLE SEARCH TOOL

2.1 Motivation and Challenges

We use search and recommender tools mainly for providing personalized suggestion for a task. A good recommender tool understands what a user is looking for. There have been different approaches to this like context based, keyword based or even Collaborative filtering. The main challenge is to build a search tool that is context aware. It tries to derive the overall meaning and big picture behind a search query and not just a local understanding.

2.2 Related Work

Understanding words and deriving meaning from them have been researched since 1980's. The earliest significant work on word representation started with Brown, *et al* on bigram and trigram models of predicting the next word(s) given the previous words [1]. The notion "word" by itself was conceived for a machine by Elman , where he explores the correlation between commonly occurring words that form sentences (when sentences are generated at random). He used a network with 5 input units, 20 hidden units, 5 output units, and 20 context units to produce a 5-bit representation for the words [2]. When Rumelhart, *et al* perfected the backpropagation algorithm for neural networks, word representations started to form a much more concrete structure and the tasks for future work started to get more clear [3].

We have now come a long way from bigram and trigram models to learn an n-gram representation. In any case, the straightforward procedures are at their limits in numerous errands. For example, there are limited amounts of relevant in-domain information for automated speech recognition whose performance is typically controlled by the size of high-quality transcribed speech data (often just millions of words). In machine translation, the existing corpora for numerous dialects contain as it were a couple of billions of words or less. Mikolov ,*et al* gave the idea computing continuous vector representations of words from very large data sets [4]. The quality of these representations

were measured in a word similarity task. His work on word2vec has been an industry standard in a lot of NLP practices and served as one of the first steps in semantic cognition. It sparked a lot of applications in Information Retreival, Document Classification and many more. Pennington, *et al* suggested that two words can be better distinguished with respect to a context word by the ratio of the co-occurrence probabilities [5]. This function, that distinguishes two word vectors based on a context word vector need to be formalized in a way that it is symmetric between context words and words. Their model weighs all the co-occurrences equally. This GloVe vector representation has taken the best of the Matrix factorization models and Local context window models and presented a computationally simple yet efficient model to determine a rich distributed word representations.

Text classification and clustering have always played an important role in document retrieval. Such algorithms typically require a fixed-length vector to represent the text input. Le , *et al* proposed the idea of a paragraph vector, an unsupervised framework that learns continuous distributed vector representations for pieces of texts [6]. The texts can be of variable-length, ranging from sentences to documents. This technique was inspired from a multiple researches (Bengio, *et al* [7] , Collobert , *et al* [8]) where each word is represented by a vector which is concatenated or averaged with other word vectors in a context, and the resulting vector is used to predict other words in the context. Paragraph vectors attempt to learns features that are relevant to the tasks at hand given very limited prior knowledge.

Over time, learning representations were explored using transfer learning based approaches, graph based approaches and using convolutional neural networks. Koopman, *et al* used a 2 step approach to improve discriminating ability of the vector representations for document embeddings [9]. They first reduce the dimensionality of the term vectors by an extremely efficient implementation of random projection [10] [11]. They then project the term vectors on the hyperplane orthogonal to the average language vector. Tang, *et al* studied the problem of integrating very large

information networks into low-dimensional vector spaces, which is useful in many tasks such as visualization, classification of nodes and prediction of links [12]. The authors proposed a model that optimizes an objective which preserves both the local and global network structures. In many real world scenarios of graph based embedding algorithms [13] [14] [15], aim to preserve the *first-order* proximity between the vertices. However, these are seldom sufficient. In the paper, the authors explore a *second-order* proximity between the vertices, defined not by the observed intensity of the connection, but by the mutual neighborhood structures of the vertices. Peters, *et al* [16] and Conneau, *et al* [17] used bi-LSTM based learning representations that model dynamic word-use characteristics and how they vary across various linguistic contexts.

2.2.1 Working of paragraph vectors

Before we get to paragraph vectors, let us look at continuous Bag-of-words (CBOW) representation. CBOW's way of working is that it tries to estimate the probability of a word given a context. A context-setting may be a single word or a group of words. Consider the example of this sentence *The quick brown fox jumped over the lazy dog*. For ease of understanding, let us consider a sliding window of length 1. This corpus can be translated as follows into a training set for a CBOW model

Input	Output		The	Quick	Brown	Fox	Jumped	Over	The	Lazy	Dog
The	Quick		1	0	0	0	0	0	0	0	0
Quick	Brown		0	1	0	0	0	0	0	0	0
Brown	Fox		0	0	1	0	0	0	0	0	0
Fox	Jumped		0	0	0	1	0	0	0	0	0
Jumped	Over		0	0	0	0	1	0	0	0	0
Over	The		0	0	0	0	0	1	0	0	0
The	Lazy		0	0	0	0	0	0	1	0	0
Lazy	Dog		0	0	0	0	0	0	0	1	0
Dog	Quick		0	0	0	0	0	0	0	0	1

Figure 2.1: One-hot Encodings

In the image below, the matrix on the right represents the one-hot encoded data on the left.

Consider the target of one of the datapoints, say that for the term "Fox" What follows this is a

Fox	Jumped		0	0	0	1	0	0	0	0	0

Figure 2.2: Example of a random encoded vector

way to convert this one-hot encoded representation into the word embedding. This matrix displayed in the above image is sent to a three-layer shallow neural network: an input layer, a hidden layer and an output layer. The output layer is a softmax - used to sum the probabilities obtained in the output layer up to 1. From the Figure 2.2 , We have the encoded vector as the input to this neural network. The hidden layer is a randomly initialized set of numbers that form the feature space of the entire vocabulary. This means, we have an d-dimension vector for every word in the vocabulary. This is equivalent to telling that the hidden layer acts as a look-up table for every word. Let us for example, fix $d = 6$

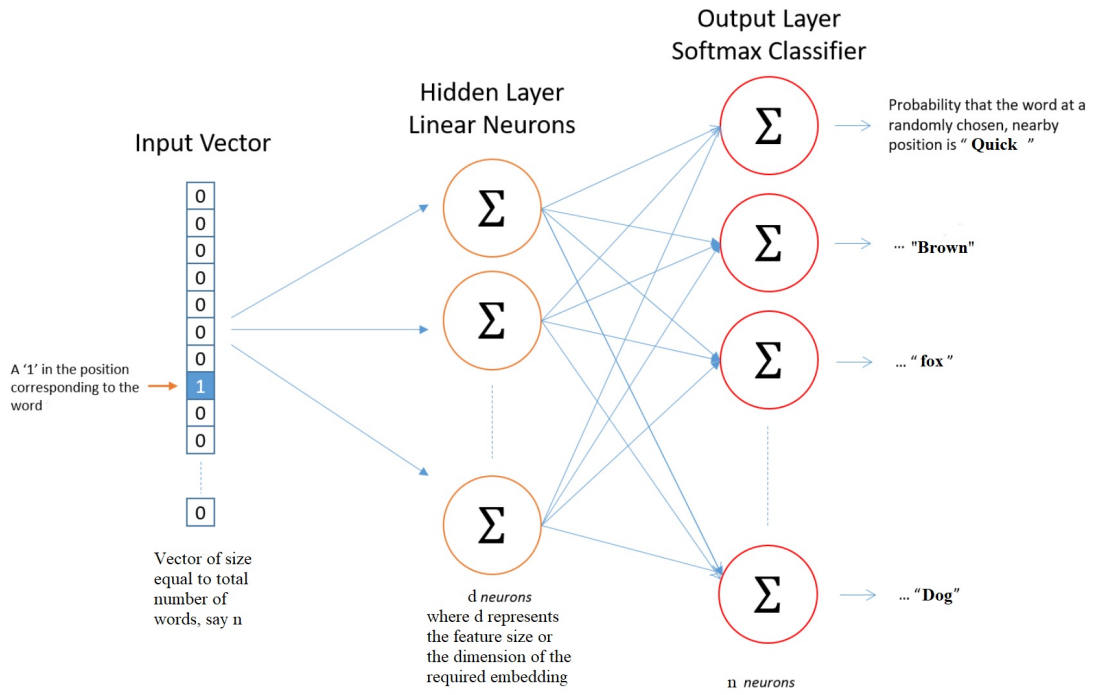


Figure 2.3: Architecture of the neural network [6]

From the above example, this can be represented as :

Figure 2.4: Weights from the example up to hidden layer

The network output is a single vector (also with 6 components) containing the probability of a randomly selected nearby word being that vocabulary word for each word in our vocabulary. Note that the hidden layer neurons do not have an activation function. Now consider the hidden layer. Looking at the rows of this weight matrix, it's really what our word vectors are going to be. So the ultimate goal of all this is only learning this matrix of hidden layer weight. Extending this example to learning 3 words, we get

Figure 2.5: Weights from the example up to hidden layer for 3 samples

The above picture takes 3 contextual words and estimates a target word's probability. As shown above in blue, orange and green, the output can be assumed to take three one-hot encoded vectors in the input layer. The measures remain the same, only the calculation of hidden activation changes. Instead of simply copying the input-hidden weight network rows to the output layer, all matrix rows are taken on average. With the above example, we can appreciate this. The output layer is a softmax regression classifier. - output neuron (one per term in the lexicon) yields between 0 and 1,

and all these output values are summed up to 1.

Next comes the training. The objective function is the negative log likelihood of a word, given its context. Here this likelihood is the softmax output of the network. The gradients are updated by the rule of stochastic gradient descent and we finally learn the weights of the words in our vocabulary. While we can train our word vectors based on our corpus, we can also use existing weights that are available online to speed up our training.

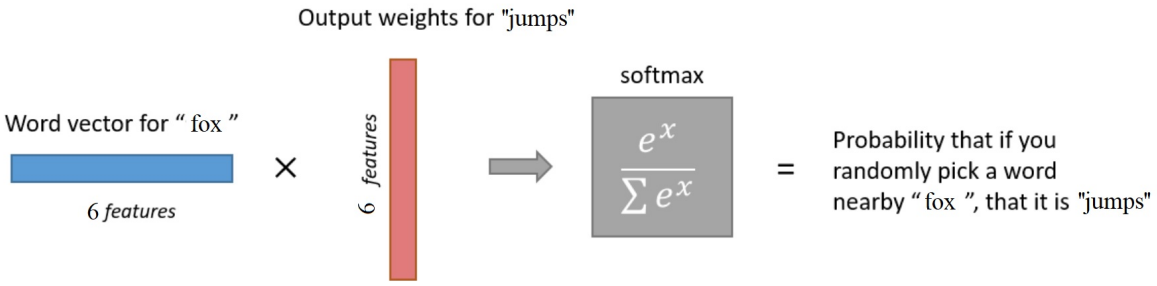


Figure 2.6: Output Layer [6]

2.2.2 Document Vectorization

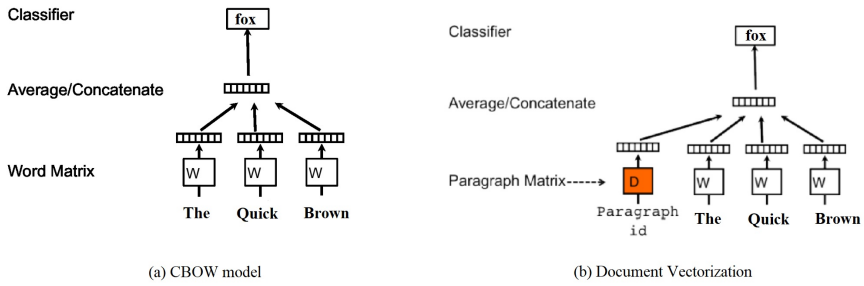
Document Vectorization is an extension to the word vectorization approach towards documents. Its intention is to encode (whole) documents, consisting of lists of sentences, rather than lists of ungrouped sentences. Let us look at some differences to understand an overview before moving to an example.

- While word vectorization calculates a feature vector for every word in the corpus, Document Vectorization computes a feature vector for every document in the corpus.

- While word vectorization works on the instinct that the word representation ought to be good enough to predict the surrounding words, the underlying intuition of Document Vectorization is that the document representation should be good enough to predict the words in the document.

- For example, word vectorization's learning strategy exploits the idea that the word *mat* follows the phrase the *cat sat on*. Document Vectorization's learning technique takes advantage of the assumption that the interpretation of adjacent terms relies unambiguously on the text. Though the appearance of the phrase *catch the ball* is common in the corpus, when we know the subject of a paper is "software" we can predict terms like *bug* or *exception* after the word *catch* (ignoring *the*) instead of the word *ball* since *catch the bug/exception* is more likely under the topic "software". On the other hand, if the topic of the text is perhaps "football" we can expect *ball* after *catch*.

Now that we are familiar with the way how CBOW works, document vectorization can be explained easily with the same principle.



(a) CBOW model (b) Document Vectorization

Figure 2.7: Comparison of CBOW and document vectors [6]

Notice that in Figure 2.7a, the CBOW representation is shown. In Figure 2.7b, we find that there is an additional paragraph matrix *D* shown that is a part of the hidden layer. Just like how

the word vectors were generated in the previous case, the document vector D is also generated. So, when training the word vectors W, the document vector D is trained as well, and in the end of training, it holds a numeric representation of the document. The rest of the procedure is very similar to the CBOW representation.

Let us go back to the same example where we consider the sentence *The quick brown fox jumped over the lazy dog* as a whole document. The document vector is also averaged with the word vectors to produce the new document vector for this sentence.

Figure 2.8: Flow of document vector generation upto hidden layer

This is once again followed by a softmax layer to obtain the probability of the next word, given the document and the context. Just like in the previous case, we need only the hidden layer weights. The contexts are fixed-length and analyzed over the article from a sliding window. The vector of a document is distributed in all contexts created from the same paragraph, but not throughout the document. However, the W word vector matrix is exchanged across paragraphs. Using stochastic gradient descent, both D and W are conditioned and the gradient is extracted by backpropagation. At prediction time, one needs to perform an *inference* step to compute the document vector for a new text input. This is also obtained by gradient descent. The document vectors are obtained by

training a neural network on the task of predicting a probability distribution of words in a document given a randomly-sampled word from the paragraph. The hyperparameter that is in our control is the embedding dimension.

2.3 Design - Building a robust search tool

In this experiment, we focus on building a robust document search tool. One inherent property of Doc2Vec model is the use of negative sampling. The theory of negative sampling was based on the concept of noise contrastive estimation (analogous, as generative adversarial networks), which states, that a good model must discern false signals from the real ones by the means of logistic regression. Also, the motivation behind negative sampling objective is similar to stochastic gradient descent: Instead of adjusting all the weights every time taking into account all the thousands of observations we have, we use only K of them and also dramatically increase computational performance. We can vary the number of negative samples. If we have to consider noise to be induced in the model, it means any we must consider any form of perturbations in the embedding space. This noise can be considered in 2 ways.

- In the first assumption, we consider words around context as noise
- In the second assumption, we consider perturbations in the embedding space itself as noise.

This is on the belief that changing the embeddings on the whole might force the model to capture nearby points that may target to some other document. For this, we induce a Gaussian noise of varying variance and plot the model accuracy. This noise is added to the embedded test sequence and passed to the initially trained noise free model to predict the probability of the required document.

2.4 Performance Evaluation

In our first assumption, we have modeled the noise as a unigram distribution, defined by:

$$U(w_i) = \frac{f(w)^{\frac{3}{4}}}{\sum_{j=0}^{n} f(w)^{\frac{3}{4}}} \tag{2.1}$$

The Figure 2.9 shows a plot of negative samples versus cosine similarity score for an arbitrary test document. Here, we observe that there is an increasing trend in model performance with increase in negative samples. The dips int he plot however, are due to the Popular Neighbor problem in the dataset.

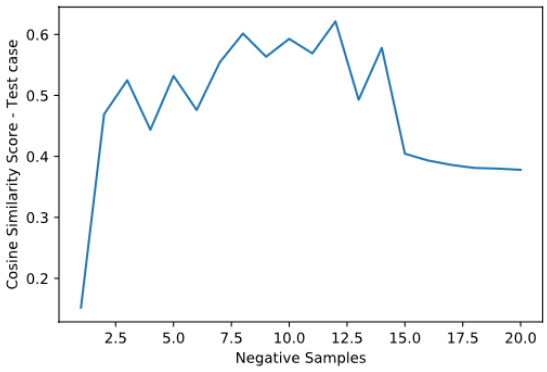


Figure 2.9: Comparing the drop in Accuracy over the 3 word count ranges

With respect to assuming the noise as a 0-mean Gaussian distribution, the Figures 2.10 and 2.11 show the prediction performance of the model with increasing standard deviation. This noise is added to the embedded test sequence and passed to the initially trained noise free model to predict the probability of the required document. Three ranges are considered here - Documents under 1000 words, between 1000 and 2000 words and over 2000 words.

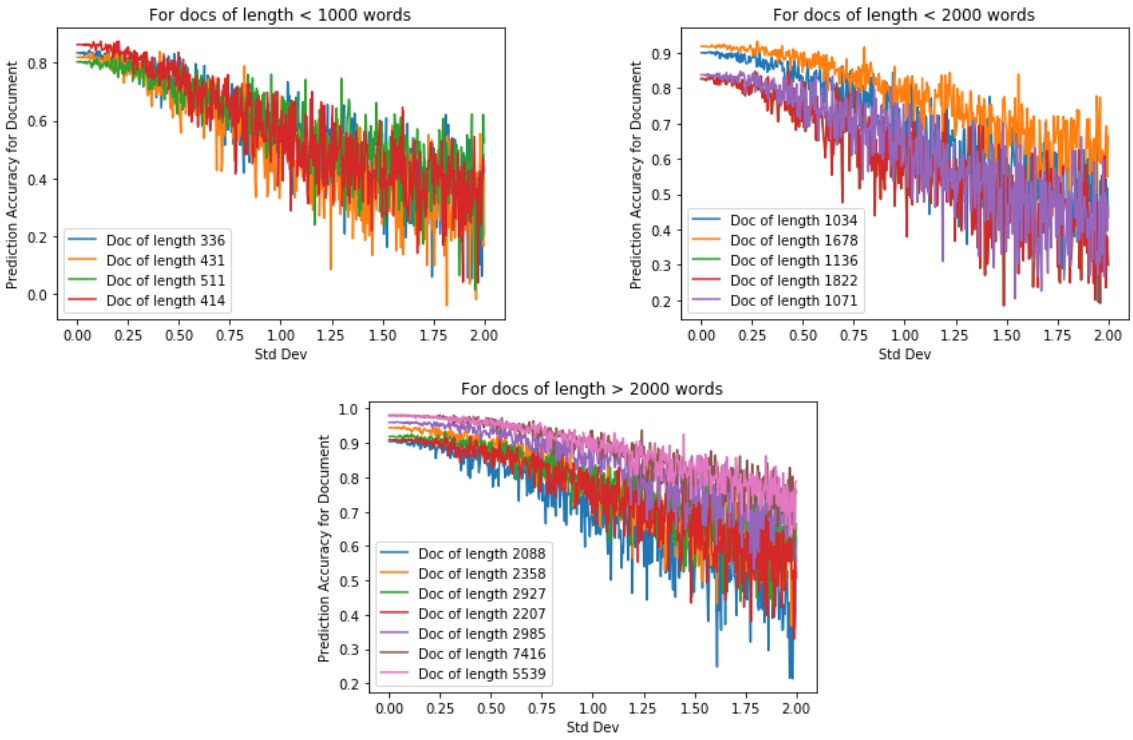


Figure 2.10: Comparing the drop in Accuracy

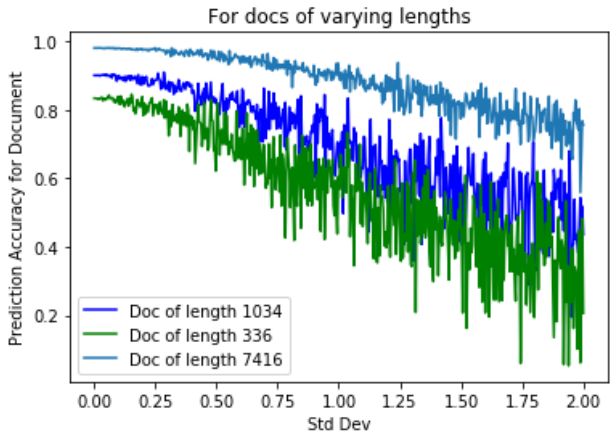


Figure 2.11: Comparing the drop in Accuracy over the 3 word count ranges

3. TOPIC CLASSIFIER

3.1 Motivation and Challenges

In the previous chapter, I presented a method to create a doc2vec model. The output of doc2vec may not be entirely satisfactory to a user because it may contain irrelevant contents that are probably “close" to the search query may show up in the results. In order to add a layer of relevance, I use a topic classifier. The challenge that may not be solved is that the term relevance is human centered and highly idiosyncratic. When a user inputs a search query and obtain a relevant “topic" for his search query, he probably was happy with the mixed bag of results the doc2vec model. There might also be a case when a user did not want the relevant topic suggested by my model. If that is the case, then the search query by itself could need revision.

3.2 Related Work

This is one of the most experimented idea in NLP. Topic Classification has been approached in a number of methods. Shallow learning methods including distributed representation were heavily employed because they could efficiently construct high-quality word embeddings and because they could be used for semantic compositionality [4] [18]. Soon, with some drawbacks with word-level embeddings, researchers delved into two sides. The first being character level embeddings and the second completely involving deep neural networks. Zhang, *et al* used a character-based representation of text as input for a convolutional neural network [19]. The approach’s hope was that if a CNN could learn to interpret the key data, all the labor-intensive work needed to clean and compose text could be resolved. Kim, *et al* was one of the first to bridge the gap between word vector embeddings and convolutional neural networks [20]. Sentences were converted to embedding vectors and made accessible to the system as a matrix output. Convolutions are performed word-wise throughout the output using kernels of different sizes, such as 2 and 3 terms at a time. In order to condense and summarize the extracted functions, the resulting function maps were then processed using a max pooling layer Zhang, *et al* performed a sensitivity analysis into the hyper-parameters

needed to configure a single layer convolutional neural network for document classification [21]. Their aim was to provide general configurations that can be used for configuring CNNs on new text classification tasks. Recently, Attention based networks have shown to get State-of-the-art results in topic classification tasks. Yang, *et al* introduced the concept of hierarchical attention (HAN) based network that employs sentence level and word level attention [22].

3.2.1 Attention Network

Attention Model, first introduced for Machine Translation [23] has now become a predominant concept in neural network literature. The intuition behind attention can be best explained using human biological systems. For example, our visual processing system tends to focus selectively on parts of the image,while ignoring other irrelevant information in a manner that can assist in perception. Let us consider a technical approach to understanding Attention Mechanism. Attention is essentially a map, often with softmax function the inputs of dense layer.Until Attention Mechanism, translating depended on reading a full sentence and compressing all data into a fixed-length vector. As you can imagine, a sentence of hundreds of words defined by several words would surely lead to knowledge loss, insufficient comprehension, etc. Be that as it may, Attention partly fixes this issue. It helps the machine translator to display all the information contained in the original sentence, then produce the correct word according to the actual term on which it operates and the context. It can even zoom in or out (focus on local or global features) for translators.

Probabilistic Language Model's main crux is to attribute a probability to a Markov Assumption for a sentence. RNN (sequence to sequence (seq2seq)) was naturally applied to model the conditional likelihood of words because of the complexity of sentences consisting of different numbers of words. The seq2seq models is normally composed of an encoder-decoder architecture, where the encoder processes the input sequence and encodes/compresses/summarizes the information into a context vector (also called as the "thought vector") of a fixed length. This representation is expected to be a good summary of the entire input sequence. The decoder is then initialized with this context vector, using which it starts generating the transformed output. A critical and

apparent disadvantage of this fixed-length context vector design is the incapability of the system to remember longer sequences. Often is has forgotten the earlier parts of the sequence once it has processed the entire the sequence. The attention mechanism was born to resolve this problem. The central idea behind Attention is not to throw away those intermediate encoder states but to utilize all the states in order to construct the context vectors required by the decoder to generate the output sequence. In brief, attention in deep learning can be loosely interpreted as a vector of meaning weights: to predict or infer an item, such as a pixel in a picture or a term in a sentence, we calculate how strongly it is associated with (or “attends to" as one may have read in many papers) other elements using the attention function. It then takes the sum of their values weighted by the attention vector as the approximation of the target.

3.2.2 Hierarchical Attention Network

When we think of the textual information nesting structure: character $\in$ word $\in$ sentence $\in$ document, a hierarchical emphasis can be built either from top-down (word-level to character-level) or from bottom-up (i.e., word-level to sentence-level) to classify hints or to collect important information both globally and locally. Two BiGRUs are used to produce semantic representation at word-level and sentence-level, respectively. To order to obtain a word-level and sentence-level encoding, a pair of hierarchical attention layers are then applied:

$$h_i^{(t)} = BiGRU(v_i^{(t)}) \tag{3.1}$$

$$v_i = \sum_t softmax(u_w^T h_i^{(t)}) \cdot h_i^{(t)} \tag{3.2}$$

$$h_i = BiGRU(v_i) \tag{3.3}$$

$$c = \sum_t softmax(u_s^T h_i) \cdot h_i \tag{3.4}$$

Here, $h_i^{(t)}$ and h_i stand for hidden representation for words and sentences. u_w^T and u_w are word-level and sentence-level pattern vectors to be learned during training.The final sentence-level represen-

tation c is then fed into a logistic regression layer to predict the category.

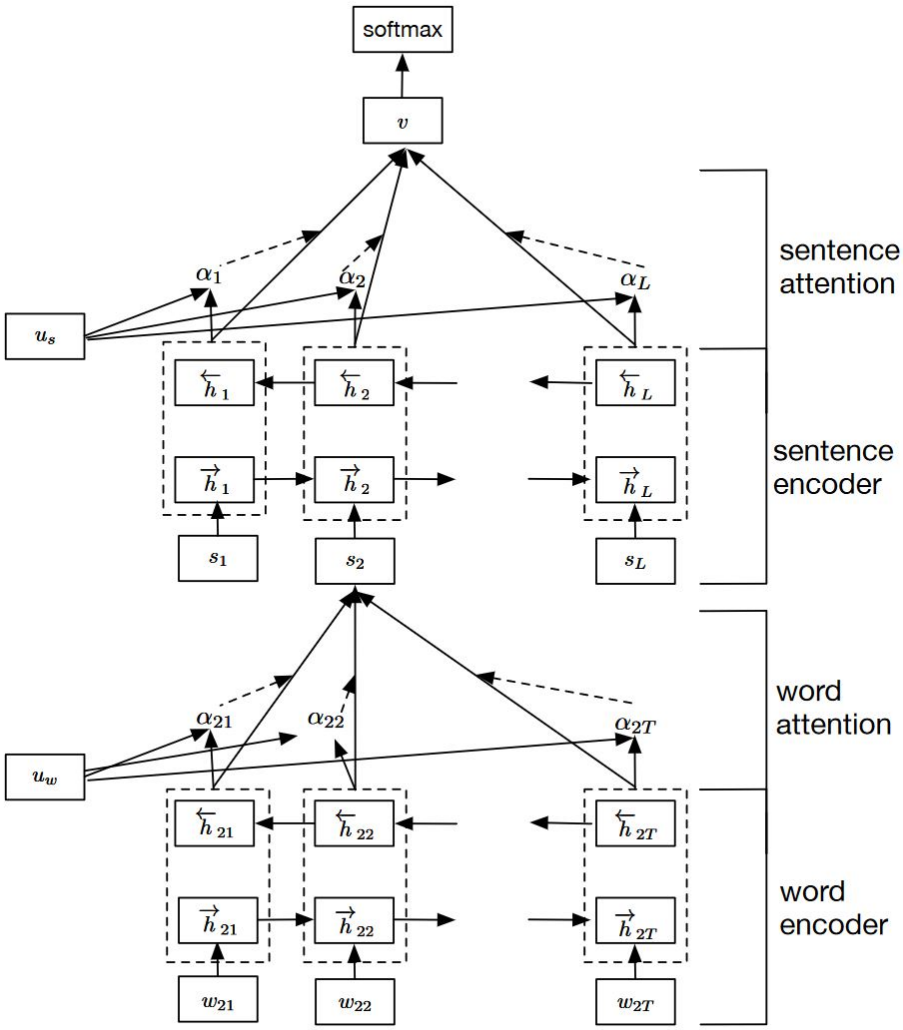


Figure 3.1: Hierarchical Attention Network [22]

We use this mechanism for document classification and predicting the class labels of documents in the dataset. The dataset used for testing this network is the DBpedia ontology classification dataset.

3.3 Designs for the Topic Classifier and Integrated Model

3.3.1 Experiment 1

We limit the number of words per sentence to 35 and the number of sentences per document to 30. A look up dictionary is created in order to encode words to numbers and we obtain a vector for each sentence and a matrix for the whole document. The model was coded to train for 10 epochs with early stopping when the loss was minimized. The training had 4375 iterations for every epoch. The batch size was set to 128. The learning rate was set to 0.1. The model converged to a training accuracy of 0.9859 with loss as 0.0488. Figure 3.3 shows the model performance for each epoch. A closer look at the iterations in an epoch is shown in Figure 3.4. The model at the end of each epoch was saved and on predicting the class labels from the testing data, we get the testing accuracy as 0.9864. The confidence interval for this model is the 70000 articles from the DBpedia set. The confusion matrix generated for this test dataset is displayed in Figure 3.2.

3.3.2 Experiment 2

In order to understand this topic classifier better, the embedding dimension was varied and the model was retrained in the same format. From the plots in Figures 3.5 - 3.8, we see that the model accuracy improves with increase in embedding dimension.

3.3.3 Experiment 3

This research focuses on combining the above mentioned applications where we explore a new method of obtaining relevant documents/texts. We analyze the input text using a Hierarchical Attention Network in order to obtain a broad category for the input text. This text, along with the newly learned topic of the text helps to obtain similar documents that are now much more relevant than spatially clustered ones. This model was run through 3 very popular datasets used in text classification.

- The AG News Dataset - The AG News collection comprises of news articles from the web-based database of news articles belonging to the four main groups (World, Business Sports,

Science / Technology). The dataset contains 30,000 training examples for each class 1,900 examples for each class for testing.

- The DBpedia Ontonlogy Dataset - The DBpedia ontology dataset contains 560,000 training samples and 70,000 testing samples for each of 14 nonoverlapping classes from DBpedia.

- Newsgroups Dataset - The 20 newsgroups dataset comprises around 18000 newsgroups posts on 20 topics split in two subsets: one for training (or development) and the other one for testing (or for performance evaluation).

For each of the cases, we trained a paragraph vector model and a topic classifier model. The results are as shown in Table 3.1 for a random test sample that generated 10 similar documents. In order to justify how the topic classifier works, we use explainable AI techniques(presented in Chapter 3) to justify why our topic classifier picked the right topic.

3.3.4 Experiment 4

In this experiment,we focus on an ensemble learning based approach to topic classifier. Ensemble learning is a broad topic and is only confined by one's own imagination. For this experiment, I have used the 20 Newsgroups dataset.

The hypothesis of ensemble averaging relies on two properties of artificial neural networks - In any network, the bias can be reduced at the cost of increased variance and In a group of networks, the variance can be reduced at no cost to bias.

3.3.4.1 Model Averaging

Every part of the ensemble contributes an equal amount to the final prediction in Model Average. A weighted ensemble is an extension of this average ensemble where each member's contribution is weighted by the model's performance. The weights of the model are small positive values and sum up to one, allowing the weights to determine the level of confidence or expected performance from each model. Uniform weight values (e.g. $\frac{1}{k}$ where k is the number of ensemble

members) implies the weighted ensemble functions as a simple average ensemble. There is no standard for calculating these weights.

3.3.4.2 Using Shallow Learning Techniques

The three most common approaches for combining multiple models ' predictions are:

- Bagging - Building multiple models from different sub-sets of the training dataset.
- Boosting - Building multiple models sequentially, where each member learns to fix the prediction errors of a prior model in the chain.
- Voting - For combining predictions several models (typically of different types) and basic statistics are used

Bagging works best with high variance algorithms. A common example is decision trees, usually constructed without pruning. In this example, we will explore a Bagged Decision Tree with 100 estimators. A 10 fold cross validation is employed. Random forest is an example of bagged decision trees. Learning information are replicated for substitution, but the trees are constructed in a way that reduces the association between individual classifiers. In general, instead of greedily selecting the best split point in tree building, only a random subset of features are regarded for each split.

AdaBoost may have been the first effective boosting ensemble. algorithm. This operates by weighing cases in the dataset by ordering the classification complexity, allowing the algorithm to give more or less attention to them as subsequent models are created. One of the most advanced ensemble techniques is stochastic gradient boosting (also called gradient boosting machines). It is a methodology that proves to be perhaps one of the best available techniques to improve performance by ensembles.

Voting is one of the easiest ways to combine the results of many machine learning algorithms.

It essentially creates two or more different models from your testing dataset first. At that stage, a Voting Classifier is utilized to wrap the models and aggregate the sub-model predictions if asked to predict new data.

3.4 Performance Evaluation

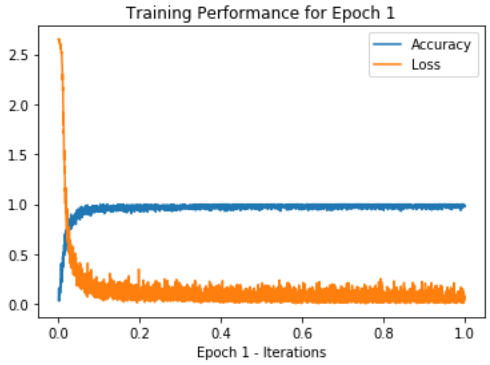
3.4.1 Results for Experiment 1

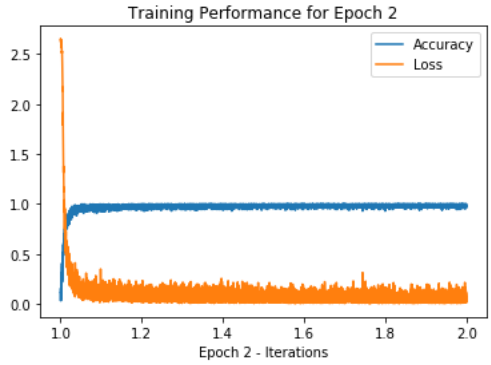
The confusion matrix generated for the test cases in the DBpedia dataset are shown below. The

```
[[4831   37    8    2    5   24   30    2    0    2    0    4    2   53]
 [  36 4931    0    0    3    0   24    0    1    0    0    0    1    4]
 [  10    0 4891   11   63    0    1    0    0    0    0    6    4   14]
 [   2    1   21 4953   19    0    0    1    0    1    0    0    0    2]
 [   1    4   72    7 4908    1    1    0    0    0    0    1    1    4]
 [  32    0    0    0    1 4954    9    0    0    0    0    0    0    4]
 [  55   38    1    0    3   12 4864   14    4    1    0    0    1    7]
 [   1    0    1    0    0    1   13 4972    5    1    1    1    2    2]
 [   1    1    0    0    0    0    1    5 4991    0    0    1    0    0]
 [   2    0    0    0    0    0    0    1    0 4975   20    0    0    2]
 [  13    0    1    0    1    1    0    1    0   27 4953    0    1    2]
 [   4    0    3    1    0    0    0    0    0    0    0 4956   21   15]
 [  10    0    2    0    0    0    0    0    0    0    0   20 4929   39]
 [  15    2    9    0    3    1    2    0    0    1    0    4   20 4943]]
```

Figure 3.2: Confusion Matrix generated for the test dataset

plots that follow show the epoch-wise iteration for the classifier model.





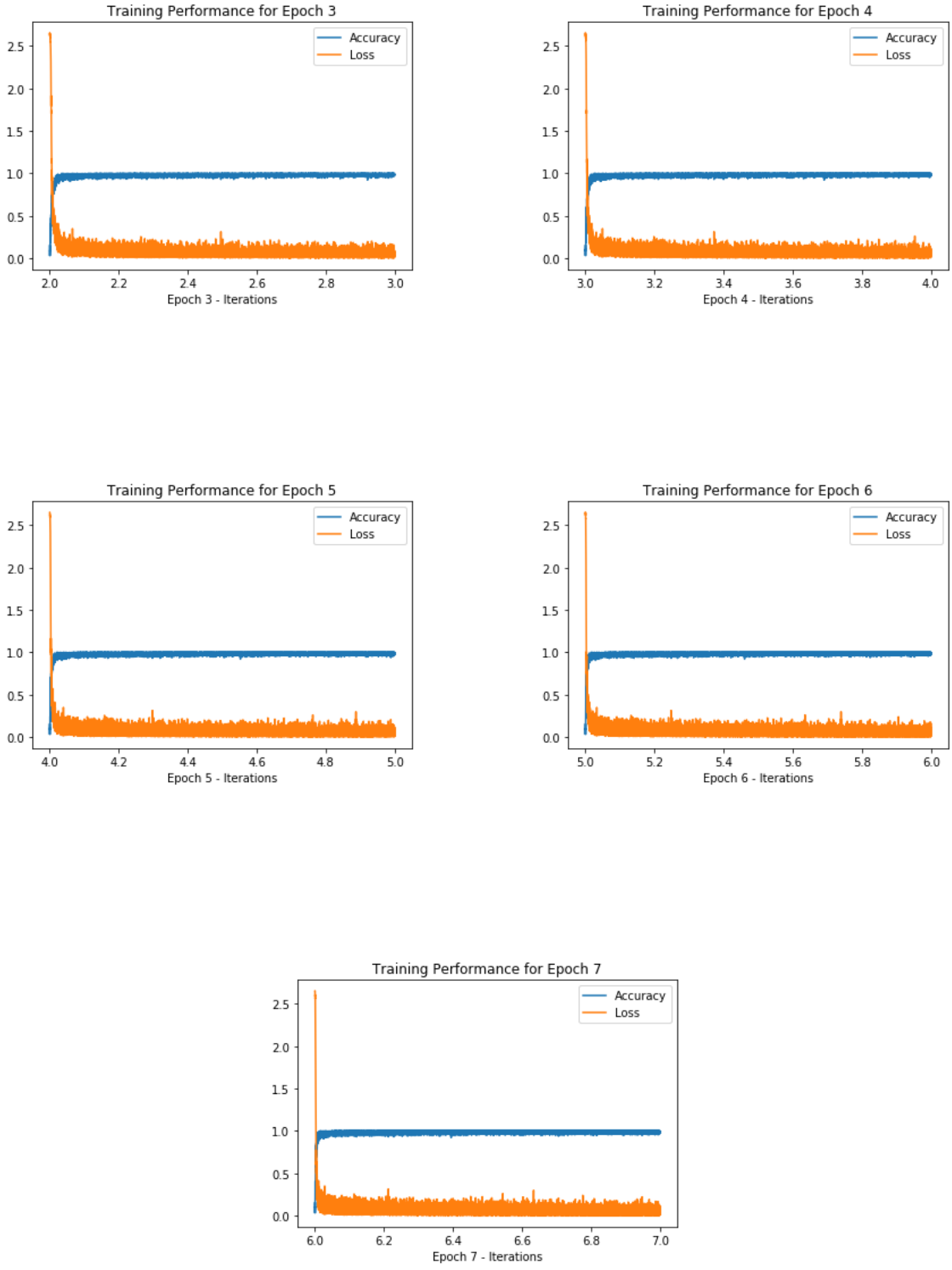


Figure 3.3: Training Performance over 7 epochs

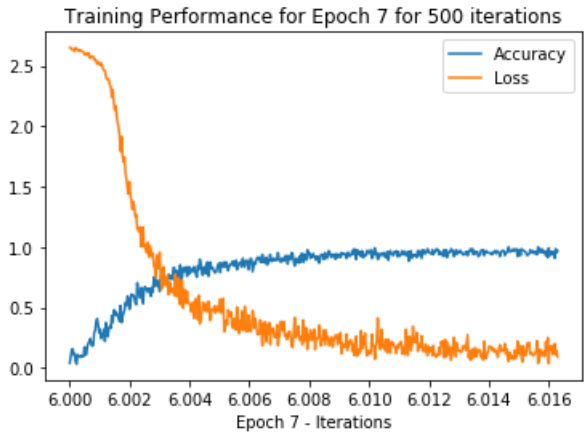


Figure 3.4: A closer look at the model performance in epoch 7

3.4.2 Results for Experiment 2

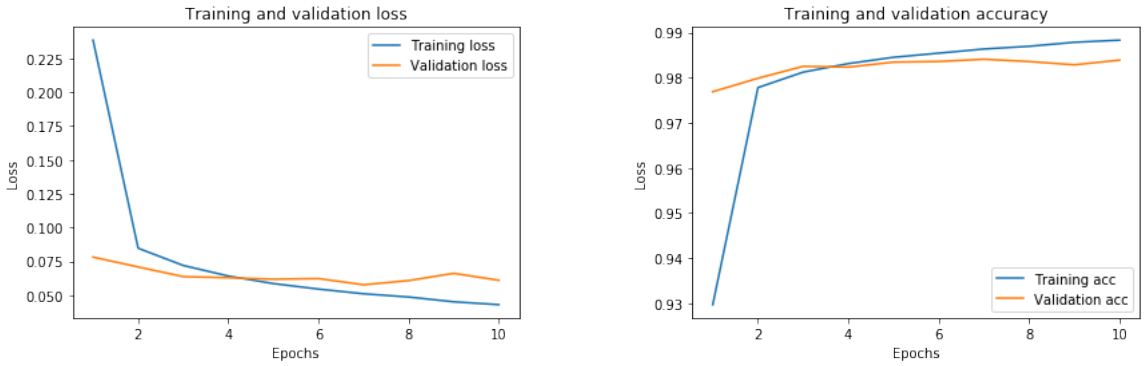


Figure 3.5: Embedding Dimension 50, Number of features = 200,000

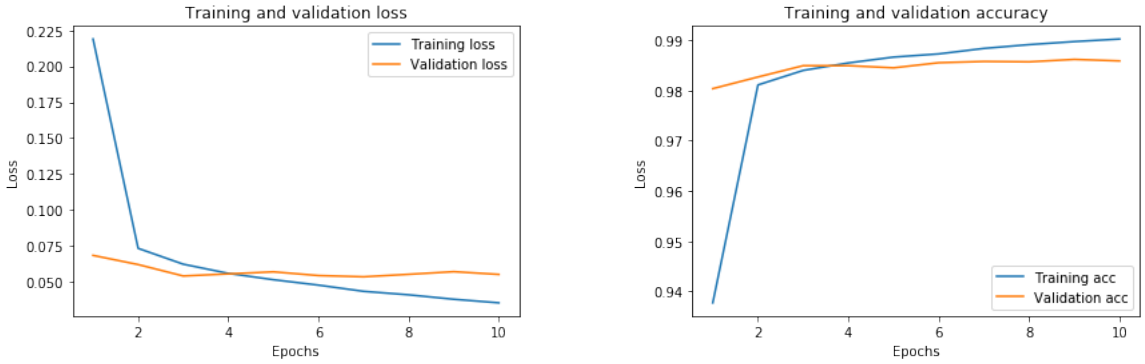


Figure 3.6: Embedding Dimension 100, Number of features = 200,000

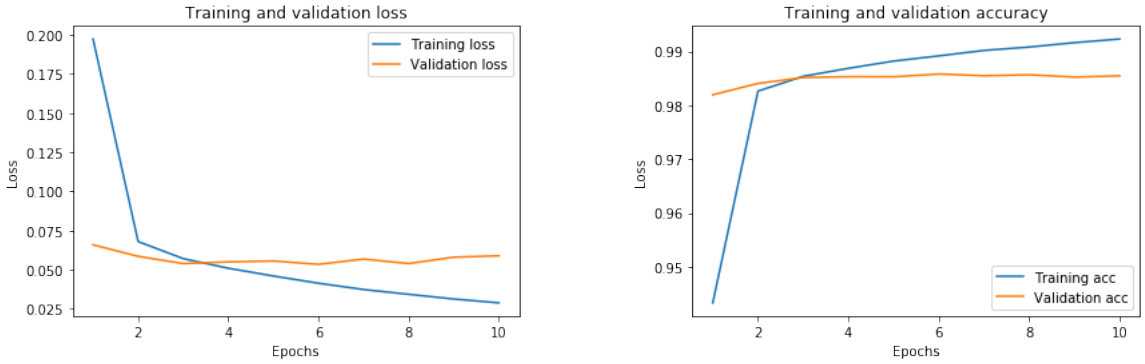


Figure 3.7: Embedding Dimension 200, Number of features = 200,000

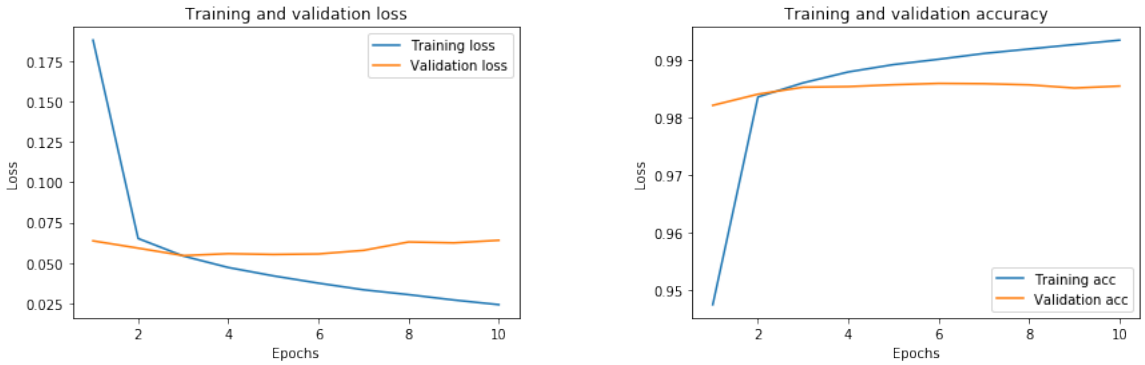


Figure 3.8: Embedding Dimension 300, Number of features = 200,000

The trend for performance plot is shown as a comparison in the Figure 3.9. We see a clear increase in the prediction performance as we increase the embedding dimension.

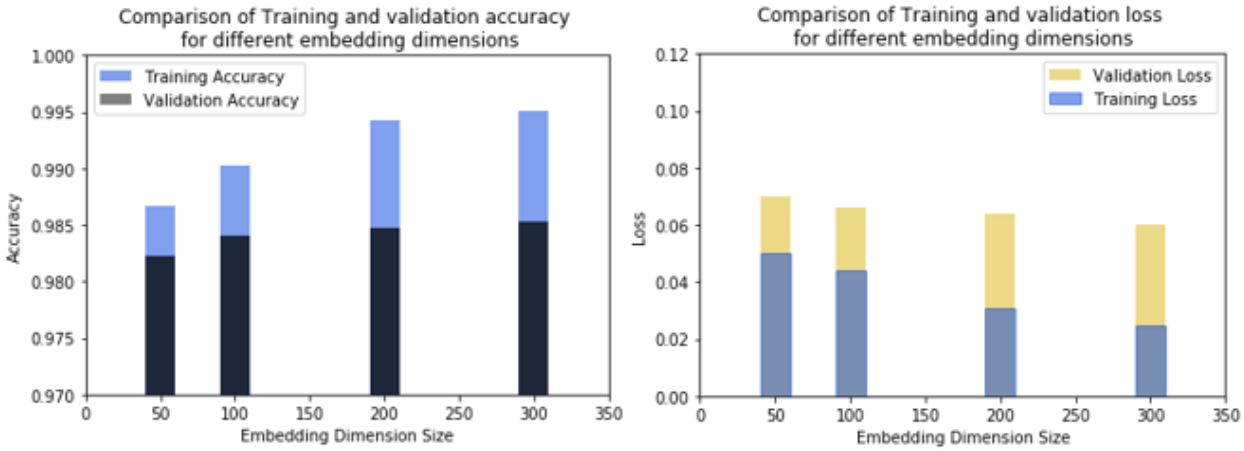


Figure 3.9: Comparison of the performance of the embedding dimension variants

3.4.3 Results for Experiment 3

From the table below, we see a significant improvement in the prediction of the article search tool when it employs the topic classifier.

AG News		DBpedia		Newsgroups	
Cosine Similarity score	**Cosine Similarity score with topic classifier**	**Cosine Similarity score**	**Cosine Similarity score with topic classifier**	**Cosine Similarity score**	**Cosine Similarity score with topic classifier**
0.4918	0.6118	0.7214	0.8516	0.6017	0.7452
0.4903	0.5877	0.7156	0.7910	0.6013	0.6988
0.4902	0.5797	0.7102	0.7843	0.5982	0.6872
0.4856	0.5634	0.6817	0.7612	0.5881	0.6776
0.4833	0.5612	0.6624	0.7500	0.5582	0.6211
0.4611	0.5499	0.6391	0.6921	0.5521	0.6019
0.4207	0.5435	0.5801	0.6345	0.5517	0.6008
0.4002	0.5388	0.5789	0.6257	0.5499	0.5912
0.3844	0.5369	0.5721	0.5912	0.5112	0.5910
0.3844	0.5364	0.5721	0.5816	0.4998	0.5800

Table 3.1: Cosine similarity score for the top 10 generated documents with and without the topic classifier

Plotting this table, we see a comparison for the 3 datasets as shown in Figure 3.10.

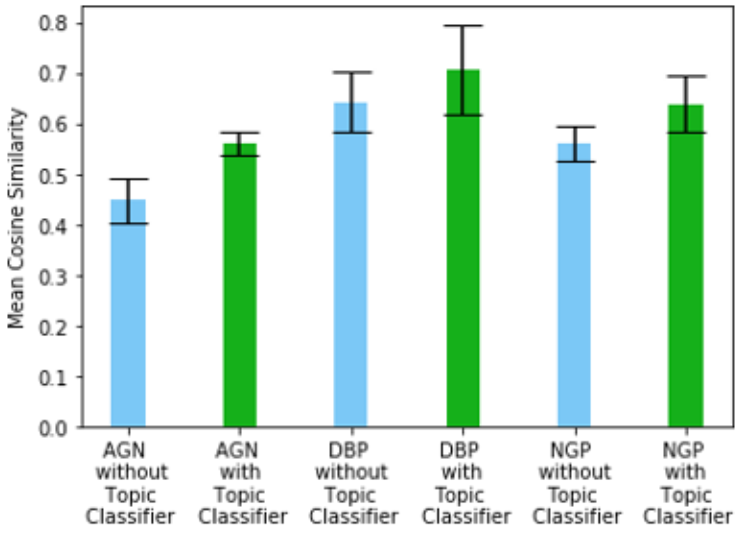


Figure 3.10: Comparison of Prediction Accuracy of the Integrated model and Doc2vec model for the 3 datasets

3.4.3.1 Query Generation

In order to generate search queries for evaluation, I removed 2-3 random words from a sentence in a document and replace them with their synonyms. I then observe if my model can capture the same document.

For example, if the document “Sentence with all alphabets" has a sentence *The quick brown fox jumped over the lazy dog*, my generated query will look like *The rapid fox over lazy dog*

3.4.3.2 Evaluation of the Integrated model

While so far, we have dealt with this problem as an unsupervised task, we can make it a supervised task in the following manner

- For each of the document in the database, I created an input-label pair of Document and its title.
- Instead of splitting the document into train and test, I will train it on the whole dataset with the words already present in the document and test on a subset of the dataset using the search query for each.
- For each of the document, I will pick a sentence at random and create a test query as mentioned above.
- Predict the set of documents.
- If my test document is present in the output set of documents with a cosine similarity score greater than a threshold value, I will count that as a correct prediction.

The above procedure will help to evaluate my approach in a supervised setting. In order to show why using my integrated model is better, the same approach is followed. The following are the evaluation results.

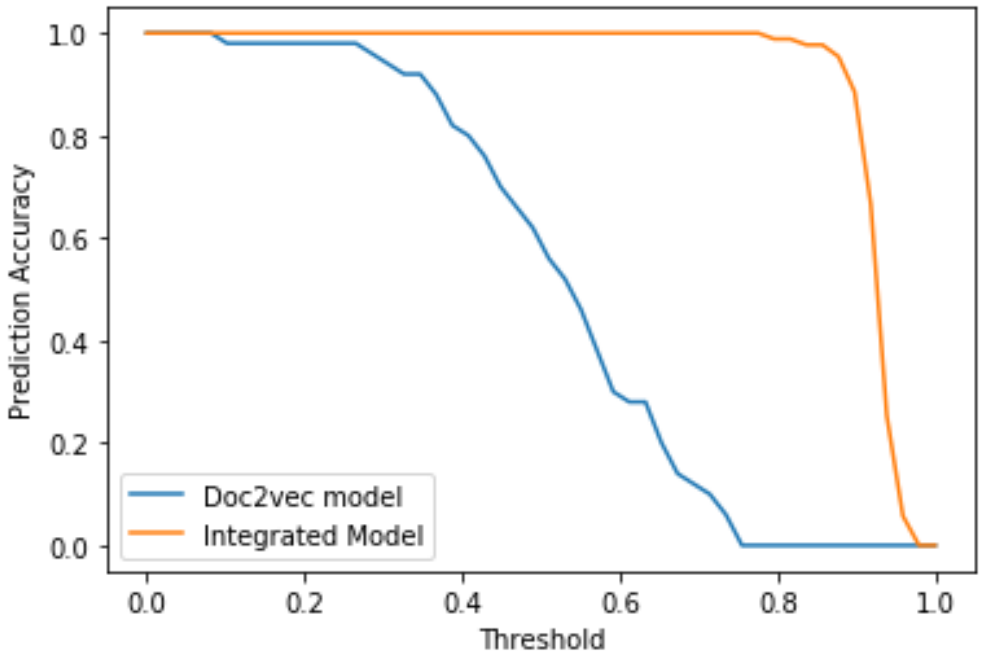


Figure 3.11: Comparison of Prediction Accuracy of the Model with respect to Threshold for prediction

The figure 3.11 shows a comparison of the prediction accuracy for arbitrarily generated test queries for 140,000 documents in the DBpedia dataset. The integrated model fares better than the doc2vec. We see that the documents can be predicted accurately in comparison to just the doc2vec model. In this plot, I varied the threshold for considering as a prediction. This means that when the cosine similarity for the document is less than the threshold, I label it as an incorrect prediction. The doc2vec model's performance drops sharply over threshold values greater than 0.5. This indicates that the doc2vec model returns documents with distance roughly around this threshold value. This is not observed in the integrated model because the model predicts documents almost surely for a threshold of 0.79 and reduces after 0.93 threshold. This value will never be equal to 1 because the my query string is not the same as the strings in my document.

3.4.3.3 Discussion

In the analysis above, I mentioned that I generate the queries for each document and observe the prediction based on a threshold. In order to validate my approach, let us look at how the true labels are created for this supervised task. Since my end goal is to predict the document, I will label all documents as class 1. At the prediction time, based on the observed value and threshold, I will assign the predicted labels to each of the documents in my test cases. Now, in a general scenario, the ground truth is given to us already and it does not depend on the task/method that follows. However, in this setting, I try to induce a supervised task setting and hence my ground truth for all documents is class 1.

3.4.3.4 Why my topic classifier helps in my integrated model

Topic Classifier adds relevance to my search query. Doc2vec model alone outputs documents that are generic and uses a distance metric. Now the question automatically becomes what is relevance? The definition for relevance in Wikipedia is *Relevance is the concept of one topic being connected to another topic in a way that makes it useful to consider the second topic when considering the first*. Relevance has proven to be a critical yet difficult term in formal analysis. Any problem requires the prior definition of the underlying principles from which a solution can be constructed. It is enigmatic, because in ordinary coherent structures, the sense of relevance tends to be problematic and incomprehensible.

Let us try to visualize this. Consider an event where you ask a friend to fetch an object from your house. Now let us also assume that this friend has never been inside your house. When you simply give this person the task and not provide any other information, he will go through the entire house till he finds it. Now, if we provide this person an extra piece of information that this object is near the study table, the person's search space is lesser and he is likely to find the object faster. This is exactly what I aim to do in my integrated model.

3.4.4 Results for Experiment 4

3.4.4.1 Model Averaging - Using only fully connected layers

We extract the features from the newsgroups dataset. The neural network used has 1 hidden layer with 100 nodes, ReLU activated and the output layer has 20 nodes. The model is fit for 25 epochs. An ensemble is created with 10 members. We have the following result, comparing the performance of the classifier with and without the ensemble.

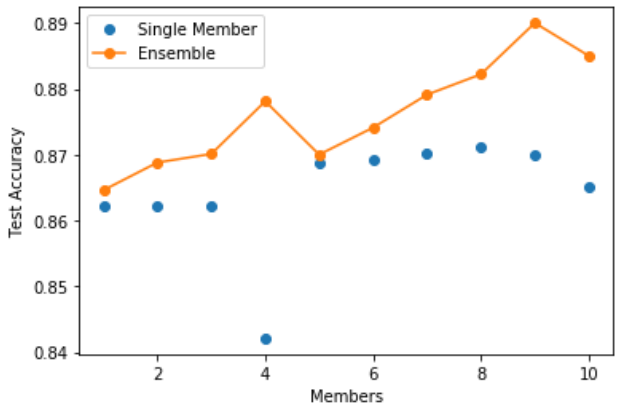


Figure 3.12: Single model versus Ensembles

3.4.4.2 Model Averaging - Using HAN

While this is the state of the art method for topic classifiers, there was not much increase in prediction accuracy in using the HAN with model averaging technique. The method remained same as what was in the previous case. We have the following result:

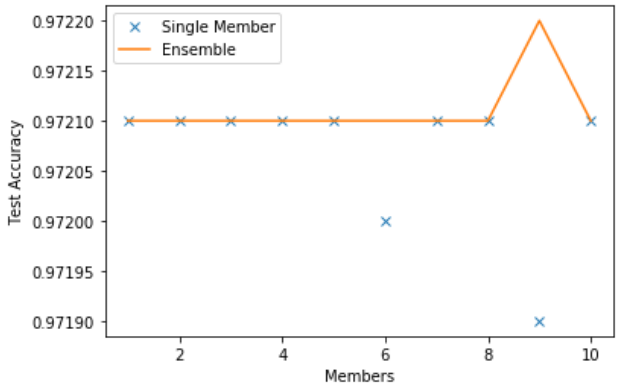


Figure 3.13: Single model versus Ensembles

3.4.4.3 Shallow Learning approach

We have the following results for the shallow learning methods on the 20 Newsgroups dataset.

Method	**Num Trees**	**Mean Test Accuracy**	**Training Time (approx)**
Adaboost	200	0.912	4 hours
Bagged Decision Trees	100	0.8876	7 hours
Random Forest	200	0.8908	4 hours
Gradient Boosting	100	0.7126	2 hours
Voting Ensemble	NA	0.8911	4 hours

Table 3.2: Mean Test performance of commonly used ensemble techniques

3.5 Applications

While I have shown a few examples where this topic classifier can be applied, there are various other areas where the model can be applied. I introduce the same model with an additional Batch Normalization layer for the sentence encoder and present 3 more examples to show the effect of Embedding variants of the HAN model. For this particular example alone, I will show the comparison of the techniques using XAI in the next section.

3.5.1 Medical Text Classifier (MTC)

In this publicly available dataset, we have around 10,000 abstracts for training, 1400 for validation and 2800 for testing. It is important to understand that these are medical datasets and it is highly likely that the labels can overlap to a large extent. Since we have 5 classes, the baseline prediction accuracy is 0.20, while our variants perform far better. The table below shows the model performance values. The 5 classes in this dataset are Neoplastic Diseases, Digestive System Diseases, Nervous System Diseases, Cardiovascular Diseases and General Pathological Conditions. The results below show the model performance for the 4 different embedding variants.

Embedding Size	Train_acc	Train_loss	Test_acc	Test_loss	Epochs
50	0.7012	0.7092	0.6014	0.9263	11
100	0.7085	0.7129	0.5959	0.9903	9
200	0.7197	0.6909	0.5872	1.0282	8
300	0.7282	0.6712	0.6049	1.0001	7

Table 3.3: MTC - Comparison of the model performance with embedding size

3.5.2 Web of Science Dataset

Web of Science (previously known as Web of Knowledge) [24] is a website which provides subscription-based access to multiple databases that provide comprehensive citation data for many different academic disciplines. It has 7 major classes namely Computer Science, Electrical Engineering, Psychology, Mechanical Engineering, Civil Engineering, Medical Science and Biochemistry. The results below show the model performance for the 4 different embedding variants. From this dataset, we have around 28,000 abstracts for training, 10,000 for validation and 10,000 for testing.

Embedding Size	**Train_acc**	**Train_loss**	**Test_acc**	**Test_loss**	Epochs
50	0.8960	0.2975	0.8508	0.4825	19
100	0.8885	0.3165	0.8525	0.4751	16
200	0.9223	0.2391	0.8947	0.3118	8
300	0.9318	0.2043	0.9018	0.2918	7

Table 3.4: WoS - Comparison of the model performance with embedding size

3.5.3 Kaggle Cancer Gene Mutation Dataset

Memorial Sloan Kettering Cancer Center (MSKCC) launched a kaggle competition to classify genetic mutations based on clinical evidence (text). There are nine different classes a genetic mutation can be classified on (Labeled from 1-9). The results below show the model performance for the 4 different embedding variants.

Embedding Size	**Train_acc**	**Train_loss**	**Test_acc**	**Test_loss**	Epochs
50	0.4219	1.2975	0.3508	1.7825	35
100	0.4247	1.4876	0.4019	1.7000	26
200	0.4621	1.2284	0.4647	1.6118	16
300	0.4718	1.2043	0.4718	1.3918	16

Table 3.5: CGM - Comparison of the model performance with embedding size

From the above results, if we plot the table values (Figure 3.14), we see that the increase in embedding dimension once again shows an increase trend in performance of the HAN model.

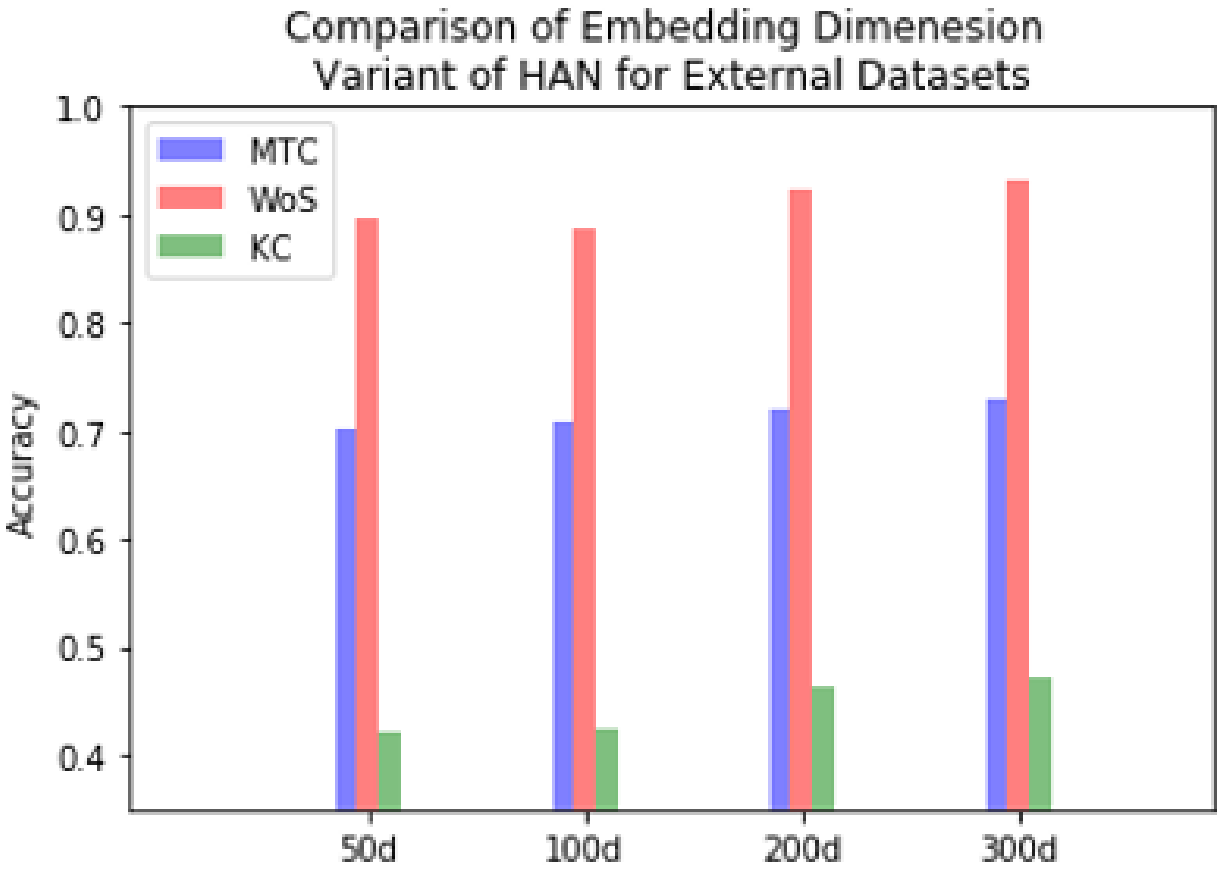


Figure 3.14: Comparison of Embedding dimension variant for the applications

4. EXPLAINABLE AI TECHNIQUES

4.1 Motivation and Challenges

The main drawback of black-box models, as the name suggests, is its inability to explain the underlying computation. While they can be very powerful for predictive analysis, it is important to know why they work. We blindly trust the recommendations of friends of Facebook, movie recommendations on Netflix and Amazon. However, we do not have the same level of trust when it comes to medical applications or actions involving life and property. This challenge is overcome by using Explainable AI.

4.2 Related Works

We divide the strategies into 3 categories -

- based on complexity of interpretability
- based on grasp of interpretability
- based on level of dependency from the learning model in that problem.

4.2.1 Complexity related strategy

A machine learning model's sophistication is directly linked to its interpretability. Generally speaking, the more complicated the model, the harder it is to understand and justify. Rule Lists and Decision trees are the most common intrinsic strategies that support the idea of making the model itself interpretable [25] [26]. However, as mentioned above, this level of interpretability trades off with the accuracy of the model. While some specific tasks in the industry involve a small number of data point (where experiments to generate data points are expensive), such model specific methods can be employed. In deep learning sense, there have been some authors who have used the attention mechanism it self to interpret the model behavior [27] [28], many have given counter examples to show merely attention is not a fail safe method for model interpretability [29] [30].

4.2.2 Interpretability via Scoop

This strategy supports two methods - comprehension of the whole model action (Global Interpretability) or a particular instance (Local Interpretability). Global interpretability enables the interpretation of a model's whole meaning and supports the whole theory leading to all the various possible consequences. This can be attributed to how the features affect class variables. While a significant number of methods are used in literature to allow global interpretability [31][32][**?**], it is arguably difficult to achieve global model interpretability in reality, especially for models that exceed a limited set of parameters. Local interpretability can be more readily applicable to humans who focus attention on only part of the system to comprehend the whole of it. So, the aim here is to generate an individual explanation, generally, to legitimize why the model made a particular decision for an instance. While there are many locally interpretable models [34] [35], we have 2 major contributions in this area - LIME [36] and SHAP [37].

4.2.2.1 LIME

The local explanation method LIME interprets an individual prediction by learning an interpretable model locally. The intuition behind LIME is that it samples instances both in the vicinity and far away from the interpretable representation of the original input. Then LIME takes the interpretable representation of these sample points, determines their predictions and builds a weighted linear model by minimizing the loss and complexity. The samples weighting is based on their distances from the original point. The points weights decrease as the points get farther away. The explanation is locally faithful, which means it represents the model prediction of vicinity instances. Let us look at this intuition mathematically.

Let f denote the complex model being explained. $f(x)$ is the probability that an instance x belongs to the certain class. Let g be defined as explanation model $g \in G$, where G is a class of

interpretable models, such as linear models and decision trees. If we assume a linear model,

$$g(z') = w_g \cdot z' \tag{4.1}$$

Also, $x \in \mathcal{R}^N$ where N is the number of features in the original input. A mapping function is defined to convert the binary pattern of missing features represented by x' to its original feature value

$$x = h_x(x') \tag{4.2}$$

LIME draws samples around x' uniformly at random. Given a perturbed sample z', LIME can recover the sample to its original input z and obtain $f(z)$, which is used as a label for the explanation model. As we know, LIME generates the explanation ξ by minimizing the loss

$$\xi(x) = \underset{g \in G}{\operatorname{argmin}} \quad \mathcal{L}(f, g, \pi_x) + \Omega(g) \tag{4.3}$$

The first term measures how unfaithful g is in approximating f. The second term is the complexity of the model. The proximity measure between x and z is denoted as an exponential kernel π_x. The loss $\mathcal{L}$ is computed by weighted square loss where $\pi_x(z)$. is an exponential kernel defined on some distance function with width.

$$\mathcal{L}(f, g, \pi_x) = \sum_{z, z' \in \mathcal{Z}} \pi_x(z) \Big(f(h_x(z') - g(z') \Big)^2 \tag{4.4}$$

To solve the above equations and to obtain the explanation, LIME approximates $\Omega(g)$ by selecting K features with Lasso and then learning the weights via least squares.

Now, how does LIME go over this procedure?

- Enable n times for each prediction to justify. The permutations are done by randomly expelling terms from the initial example when the product of text projections is to be explained

Based on whether or not the model uses word location, words that appear several times will be removed whether one-by-one or entirely.

- Let the black box model predict the result of all permuted cases.
- Measure the distance to the original from all permutations.
- Transfer the distance to a ranking of similarities based on score. As permutations are generated on the basis of inputs separately, comparisons are measured in various ways. The cosine similarity measure is used for text data which is the norm in text analysis (the angle difference between the two attribute vectors is effectively measured).
- Choose m best features from the permuted data to reflect the complex model's outcome. Choice of features is a full sub-field of modeling in itself and there's no silver bullet for LIME. Instead, it implements a variety of solutions to the choice of options that an user can choose from.
- Fit a simple model to the permuted data, explaining the black box model output with the m features of the permuted data weighted by its similarity to the original instance.
- Extract the weights of the function from the simple model and use them to illustrate the local behavior of the black box model.

The reliability of the model or the metric of consistency (how well the interpretable model approximates the predictions of the black box) gives us a great perspective into how accurate the interpretable model is to explain the predictions of the black box in the information instance of interest. The examples of regional surrogate models may use features other than the original model. This will be a huge advantage over other approaches when it is not possible to interpret the original features. A text classifier can rely on conceptual word embedding as attributes, but the interpretation may be dependent on a sentence's presence or absence of letters. The authors followed up on LIME by introducing another paper that fortified results from LIME [9]. It had a limitation that although we can use simple function to interpret complexity locally, it can only explain the particular

instance. It means that it may not able to fit for unseen instance. Anchors, the new strategy, uses “local region” to learn how to explain the model. The “local region” refer to a better construction of generated data set for explanation.

4.2.2.2 SHAP

Shapley value is a concept of solution in game theory. TThe aim is to use game theory to view the goal model. All features are "contributor" and attempting to predict the "game" task and the "reward" is the actual prediction minus the explanation model result. Shapley value is the average contribution of features which are predicting in different situation. In other word, it is not talking about the difference when the particular feature missed. The authors of SHAP prove in their NIPS paper that the values of Shapley are the only guarantee of reliability and continuity and that LIME is in essence a subclass of SHAP which lacks the same properties.

In order to understand SHAP values, let us look at a small example. Consider a development team. Our aim is to create a deep learning framework that allows 100 line of codes to be completed while we have 3 data scientists (L, M, N).

V(x)	**Lines of code**
L	10
M	30
N	5
L,M	50
L,N	40
M,N	35
L,M,N	100

Table 4.1: Contribution among coalition

We calculate the marginal contribution through all permutations of players.

Order	L Contribution	M Contribution	N Contribution
L,M,N	V(L) = 10	V(L,M) - V(L) = 50-10 = 40	V(L,M,N) - V(L,M) = 100-50 = 50
L,N,M	V(L) = 10	V(L,M,N) - V(L,N) = 100-40 = 60	V(L,N) - V(L) = 40-10 = 30
M,L,N	V(L,M) - V(M) = 50-30 = 20	V(M) = 30	V(L,M,N) - V(L,M) = 100-50 = 50
M,N,L	V(L,M,N) - V(M,N) = 100-35 = 65	V(M) = 30	V(M,N) - V(M) = 35-30 = 5
N,L,M	V(L,N) - V(L) = 40-10 = 30	V(L,M,N) - V(L,N) = 100-40 = 60	V(N) = 5
N,M,L	V(L,M,N) - V(M,N) = 100-35 = 65	V(M,N) - V(M) = 35-30 = 5	V(N) = 5

Table 4.2: Marginal Contribution by different orders

The shapley value is given by:

$$\phi_i(v) = \frac{1}{|N|!} \sum_R \left[v\Big(P_i^R \cup \{i\}\Big) - v\Big(P_i^R\Big) \right] \tag{4.5}$$

where

ϕ is the Shapley value,

N is the number of players,

P_i^R is the set of player with order,

$v\Big(P_i^R\Big)$ is the contribution of set of player with that order,

$v\Big(P_i^R \cup \{i\}\Big)$ is the contribution of set of player with that order and player i.

So, from Table 4.2 and the above Equation, we can compute the Shapley values for each of the players as

Player	Shapley Calculation	Shapley Value
L	(1/3!)(10+10+20+65+30+65)	33.33
M	(1/3!)(40+60+30+30+60+5)	37.5
N	(1/3!)(50+30+50+5+5+5)	24.17

Table 4.3: Shapley value for each player

These values makes sense because player N does least work individually, and hence has a much lower Shapley value compared to L and M. In our examples, it is difficult to show by hand calculations, the effect of each feature permuted with the rest. As a result, we can only interpret the importance of each feature per class label from the summary plot.

While we have seen a great amount of consistency in the reason for class prediction over all the cases, there are obviously some differences. Trivially, this is because the methods used are completely different. SHAP, as the title of the paper says is a Unified approach to model interpretability. It combines multiple methods and produces an optimized pipeline. Nevertheless, all the 3 models have issues related to efficiency. SHAP, especially, has proven to be extremely useful in interpreting tabular data. The authors have introduced KernelSHAP and DeepSHAP. Before we go over the mathematics behind these methods, let us see the 3 useful properties of shapley values.

- Property 1 - **Local Accuracy**

$$f(x) = g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x'_i \tag{4.6}$$

The explanation model $g(x')$ matches the original model $f(x)$ when $x = h_x(x')$, where $\phi_0 = f(h_x(0))$ represents the model output with all simplified inputs toggled off (i.e. missing).

- Property 2 - **Missingness**

$$x'_i = 0 \implies \phi_i = 0 \tag{4.7}$$

Missingness constrains features where $x'_i = 0$ to have no attributed impact.

- Property 3 - **Consistency** Let $f_x(z') = f(h_x(z'))$ and $z'\backslash x$ denote setting $z'_i = 0$. For any two models f and f', if

$$f'_x(z') - f'_x(z'\backslash i) \geq f_x(z') - f_x(z'\backslash i) \qquad \forall z' \in \{0,1\}^M \tag{4.8}$$

then,

$$\phi_i(f', x) \geq \phi_i(f, x) \tag{4.9}$$

This leads us back to the Equation 4.5, where we can rewrite the same as

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!} \big[f_x(z') - f_x(z'\backslash i)\big] \tag{4.10}$$

where $|z'|$ is the number of non-zero entries in z', and $z' \subseteq x'$ represents all z' vectors where the non-zero entries are a subset of the non-zero entries in x'.

These shapley values, as mentioned in the above equations and example, are the conditional expectation function of the original model, i.e the solution to Equation 4.10. Here, $f_x(z') = f(h_x(z')) = E[f(z)|z_S]$, where S is the set of non-zero indices in z'. The authors proposed DeepSHAP as a way to combine Shapley values and the DeepLIFT technique to interpret deep learning models. The summation-to-delta property in DeepLIFT states that

$$\sum_{i=1}^{n} C_{\Delta x_i \Delta o} = \Delta o \tag{4.11}$$

where $o = f(x)$ is the model output, $\Delta o = f(x) - f(r), \Delta x_i = x_i - x_r$, where r is the reference

input. Here, if we choose $\phi_i = C_{\Delta x_i \Delta o}$ and ϕ_0 as $f(r)$, we get our original idea of additive feature attribution. From the explanation of conditional distribution, if we assume model linearity and feature independence, we get

$$f(h_x(z')) = E[f(z)|z_S] \tag{4.12}$$

$$= E_{z_{\bar{S}}|z_S}\big[f(z)\big] \qquad \bar{S} \notin S \tag{4.13}$$

$$\approx E_{z_{\bar{S}}}\big[f(z)\big] \tag{4.14}$$

$$\approx f\Big(\big[z_S, E[z_{\bar{S}}]\big]\Big) \tag{4.15}$$

If we interpret the reference value in summation-to-delta equation and the above conditional distribution approximation, then DeepLIFT approximates SHAP values assuming that the input features are independent of one another and the deep model is linear. Since DeepLIFT is an additive feature attribution method that has local accuracy and missingness property, Shapley value bridges the remaining consistency property in helping us arrive at DeepSHAP.

DeepLIFT's multiplication rule is given by

$$m_{\Delta x \Delta t} = \frac{C_{\Delta x \Delta t}}{\Delta x} \tag{4.16}$$

In other words, it is the ratio of contribution of difference-from-reference to input neuron Δx to difference-from-reference to target neuron Δt to the difference-from-reference to input neuron Δx. Note the close analogy to the idea of partial derivatives where $\frac{\partial t}{\partial x}$ is the infinitesimal change in t caused by an infinitesimal change in x, divided by the infinitesimal change in x. The multiplier is similar in spirit to a partial derivative, but over finite differences instead of infinitesimal ones. Writing this multiplier in terms of shapley coefficients for backpropagation, we get the linear approximation for shapley values in terms of DeepLIFT.

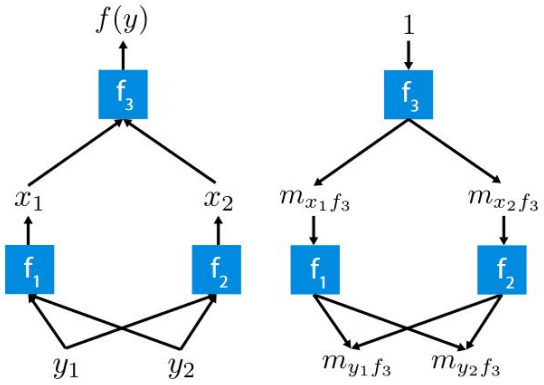


Figure 4.1: DeepSHAP interpretation [37]

$$m_{x_j f_3} = \frac{\phi_i(f_3, x)}{x_j - E[x_j]} \tag{4.17}$$

$$\forall_{j \in \{1,2\}} \qquad m_{y_i f_j} = \frac{\phi_i(f_j, y)}{y_i - E[y_i]} \tag{4.18}$$

$$m_{y_i f_3} = \sum_{j=1}^{2} m_{y_i f_j} m_{x_j f_3} \quad \text{(chain rule)} \tag{4.19}$$

This chain rule is given by

$$m_{\Delta x_i \Delta t} = \sum_{j} m_{\Delta x_i \Delta y_j} m_{\Delta y_j \Delta t} \tag{4.20}$$

where $x_1, x_2, x_3, ...$ are input neurons, $y_1, y_2, y_3, ..$ are neurons in a hidden layer and target is t.

Linearly approximating Equation 4.17, we get the shapley value for the feature as

$$\phi_i(f_3, y) \approx m_{y_i f_3}\Big(y_i - E[y_i]\Big) \tag{4.21}$$

4.2.2.3 ELI5

We also have a python package ELI5 - Explain Like I'm 5 [38]! ELI5 provides both global model interpretation and local model interpretation. The global model interpretation can be considered simply as a feature importance. It not only support decision tree algorithm such as Random Forest and XGBoost but also all scikit learn estimators. ELI5 author introduces it as Permutation Importance for global interpretation. To calculate the score, feature (a word) will be replaced by other word (noise) and predicting it. The idea is that feature importance can be deduced by getting how much score decrease when a particular word is not provided.

4.2.3 Model related methods

Another effective way to define model interpretability techniques is that if they are agnostic methods, meaning they can be extended to any form of ML algorithms, or model-specific techniques, meaning techniques that are only applicable to a particular algorithm category or class. The downside to intrinsic method practice is that when we need a particular type of analysis, we are limited to the models that provide it in terms of selection, possibly at the cost of using a more accurate and representative model. Model-agnostic representations are normally post-hoc, commonly used to explain neural networks, and could be local or global structures that can be interpreted. There are four main categories of agnostic simulation techniques -

- Visualization - Visualization techniques are applied primarily to supervised models of learning. Popular visualization techniques include: Surrogate models, Partial Dependence Plot (PDP) [39] and Individual Conditional Expectation (ICE) [40].
  - These are models that are simple to explain a complex model. In general, it is an interpretable model (like a regression model or decision tree) that is learned to approach the latter on the assumptions of the original black-box model. There are, however, almost no empirical assurances that the more complex version is strongly reflective of the basic surrogate model. The LIME methodology described above is a recommended method for constructing regional surrogate models around [36] single observations.

  - PDP - This is a visual representation that helps to illustrate the partial average association of one or more input parameters and a black-box model's predictions.
  - ICE - ICE plots extend PDP. While PD plots provide a fuzzy view of the workings of a system, ICE plots show relationships and individual differences by breaking down the PDP performance.

- This can be important inner representations when learning algorithms change cells in the hidden surface. Therefore, the function of extracting explanations from the network is to remove the knowledge acquired through learning and encoded as an internal representation[41] in a comprehensible manner.

- Influence methods - This method of techniques measures the value and importance of a function by adjusting the output or internal components and observing how the changes affect the performance of the design.

  - Sensitivity Analysis - Sensitivity refers to how the source and/or weight interference of an ANN output is affected. It is used to verify that model behavior and outputs remain stable when intentionally disrupting data or simulating other changes in data. Visualizing the results of sensitivity analysis is regarded an agnostic interpretation strategy, as the view of stability models as data shift increases confidence in the performance of machine learning [42].
  - Layer-wise Relevance Propagation - This strategy consolidates the prediction function backwards, originating from the output layer of the network and backpropagating to the input layer. The main property of this phase of restructuring is called preservation of interest. This approach discusses predictions relevant to the state of peak uncertainty [43] as opposed to sensitivity analysis.
  - Feature Importance - Variable importance assesses each input variable's contribution (feature) to a complicated ML model's predictions. The change in the prediction error of the system is determined after the feature has been permutated to measure the value

of a function. It increases the model error by allowing the values of important features. The model ignores the values of unimportant features when permuting and thus holds [44] unchanged model error.

- Example based explanation - Example based explanation techniques select particular instances of the dataset to explain the behavior of machine learning models. Example-based explanations are mostly model agnostic because they make any ML model more interpretable [45].

4.3 Designs

4.3.1 Experiment 1

Given a random test input, I generate explanations for the topic classifier model using the XAI techniques.

4.3.2 Experiment 2

In Experiment 2 in Section 2, we varied embedding dimensions for the HAN model. Here, I will generate explanations for each embedding variant to compare and show the improvement in the prediction performance.

4.3.3 Experiment 3

This experiment focuses on the idea of model tuning using XAI techniques. For this, I will generate explanations for the topic classifier as it trains.

4.3.4 Experiment 4

As mentioned in Section 3.3.2, the new HAN with a batch normalized layer has been compared with the embedding dimension variant along with the XAI techniques.

4.4 Performance Evaluation

4.4.1 Result for Experiment 1

Consider a random test input:

Input Text : Share prices for the British drugmaker Indivior plunged today on the London Stock Exchange. The drop came on news that the U.S. Justice Department indicted the company on fraud and conspiracy charges. Indivior makes the drug Suboxone, widely used to treat people suffering from opioid addiction. Federal prosecutors now claim the company falsely marketed Suboxone as safer and less prone to abuse than cheaper generic drugs. North Country Public Radio's Brian Mann reports.

The **predicted class** from the hierarchical attention network is : Company. The attention weights for the words are shown below, sentence-wise.

From these attention weights, we get the words with maximum attention values as

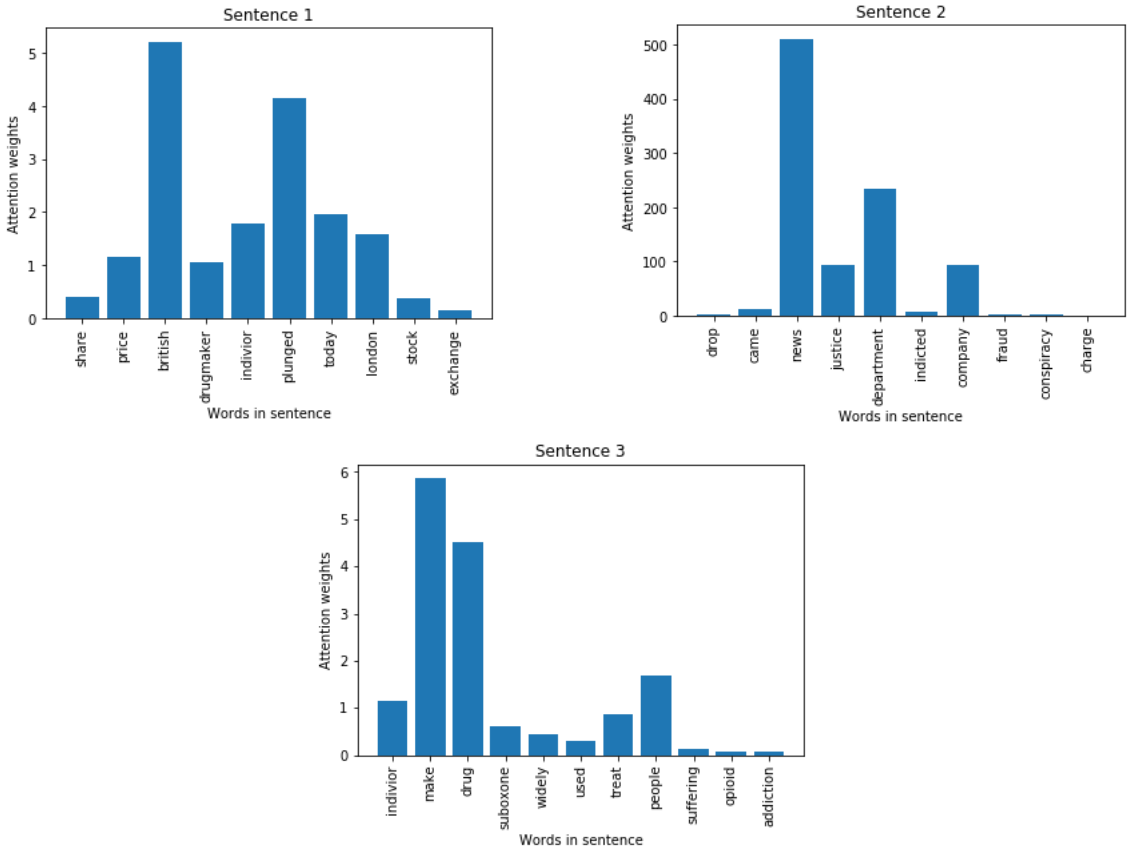


Figure 4.2: Attention weights for words in every sentence

```
Words which have highest attention weights:  ['news', 'department', 'company', 'justice', 'came', 'indicted', 'make', 'briti
sh', 'drug', 'plunged', 'conspiracy', 'fraud', 'today', 'indivior', 'people', 'london', 'drop', 'price', 'drugmaker', 'charg
e', 'treat', 'suboxone', 'widely', 'share', 'stock', 'used', 'exchange', 'suffering', 'opioid', 'addiction']
```

Figure 4.3: Attention Output with words that have max attention weights

Next we look at the output from LIME, Eli5 and Shap.

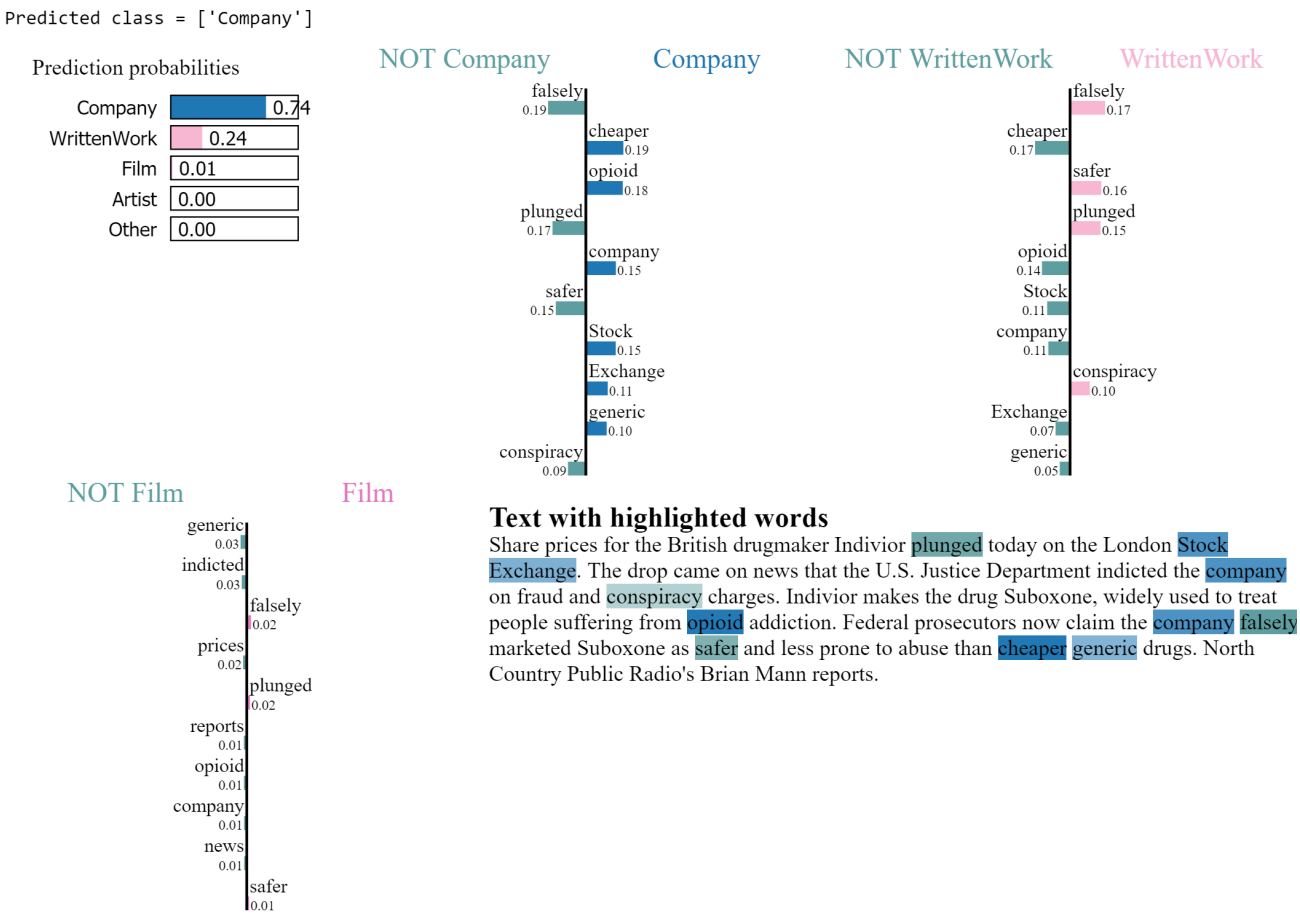


Figure 4.4: LIME output

y=Company (probability **0.685**, score **1.504**) top features

Contribution[?]	Feature
+2.043	Highlighted in text (sum)
-0.538	<BIAS>

share prices for the british drugmaker indivior plunged today on the london stock exchange. the drop came on news that the u.s. justice department indicted the company on fraud and conspiracy charges. indivior makes the drug suboxone, widely used to treat people suffering from opioid addiction. federal prosecutors now claim the company falsely marketed suboxone as safer and less prone to abuse than cheaper generic drugs. north country public radio's brian mann reports.

y=Artist (probability **0.002**, score **-6.040**) top features

Contribution[?]	Feature
-0.442	<BIAS>
-5.598	Highlighted in text (sum)

share prices for the british drugmaker indivior plunged today on the london stock exchange. the drop came on news that the u.s. justice department indicted the company on fraud and conspiracy charges. indivior makes the drug suboxone, widely used to treat people suffering from opioid addiction. federal prosecutors now claim the company falsely marketed suboxone as safer and less prone to abuse than cheaper generic drugs. north country public radio's brian mann reports.

y=WrittenWork (probability **0.301**, score **-0.578**) top features

Contribution[?]	Feature
+0.138	Highlighted in text (sum)
-0.716	<BIAS>

share prices for the british drugmaker indivior plunged today on the london stock exchange. the drop came on news that the u.s. justice department indicted the company on fraud and conspiracy charges. indivior makes the drug suboxone, widely used to treat people suffering from opioid addiction. federal prosecutors now claim the company falsely marketed suboxone as safer and less prone to abuse than cheaper generic drugs. north country public radio's brian mann reports.

y=OfficeHolder (probability **0.002**, score **-5.941**) top features

Contribution[?]	Feature
-0.431	<BIAS>
-5.510	Highlighted in text (sum)

share prices for the british drugmaker indivior plunged today on the london stock exchange. the drop came on news that the u.s. justice department indicted the company on fraud and conspiracy charges. indivior makes the drug suboxone, widely used to treat people suffering from opioid addiction. federal prosecutors now claim the company falsely marketed suboxone as safer and less prone to abuse than cheaper generic drugs. north country public radio's brian mann reports.

y=Film (probability **0.009**, score **-4.509**) top features

Contribution[?]	Feature
-0.534	<BIAS>
-3.975	Highlighted in text (sum)

share prices for the british drugmaker indivior plunged today on the london stock exchange. the drop came on news that the u.s. justice department indicted the company on fraud and conspiracy charges. indivior makes the drug suboxone, widely used to treat people suffering from opioid addiction. federal prosecutors now claim the company falsely marketed suboxone as safer and less prone to abuse than cheaper generic drugs. north country public radio's brian mann reports.

Figure 4.5: Eli5 output

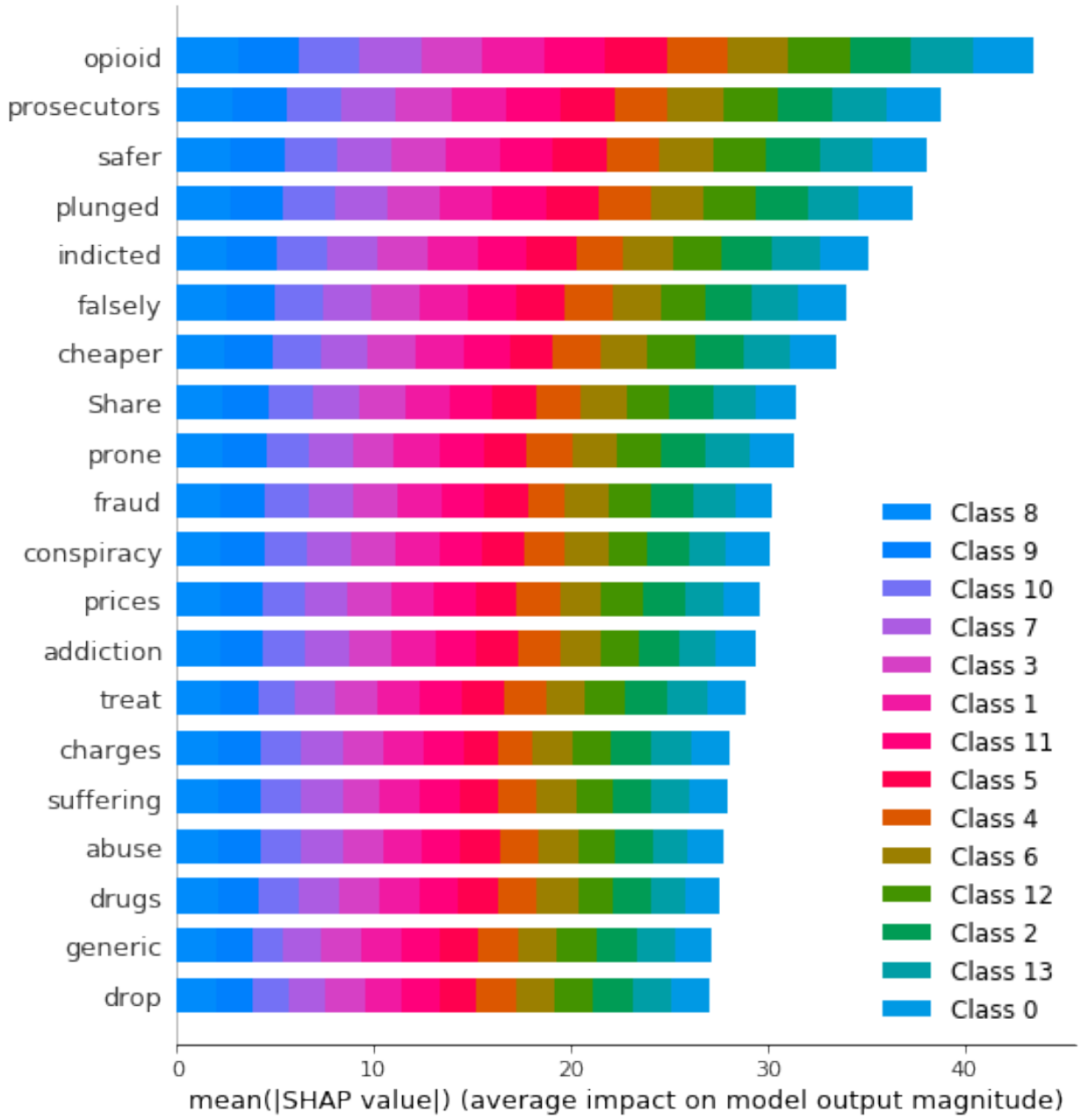


Figure 4.6: SHAP summary plot - Class labels correspond to 0: 'Company', 1: 'Educational Institution', 2: 'Artist', 3: 'Athlete', 4: 'Office Holder', 5: 'Mean Of Transportation', 6: 'Building', 7: 'Natural Place', 8: 'Village', 9: 'Animal', 10: 'Plant', 11: 'Album', 12: 'Film', 13: 'Written Work'

The table below highlights Which words are missing and which are found in common among the 4 methods discussed above. The color coding is as follows: Yellow - 2 similar words ; Light Green - 3 similar words ; Dark Green - All models have the words

Similar words and Dissimilar words			
Attention Network	**LIME**	**SHAP**	**Eli5**
News	Company	Opioid	Share
Department	Stock	Prosecutors	Price
Company	Cheaper	Safer	Stock
Justice	Opioid	Plunged	Exchange
Indicted	Generic	Indicted	Company
Came	Exchange	Falsely	Charges
Make		Cheaper	Makes
British		Share	Opioid
Drug		Prone	Fraud
Plunged		Fraud	Federal
Conspiracy		Conspiracy	Cheaper
Fraud		Prices	Generic
Today		Addiction	Marketed
People		Treat	
London		Charges	
Price		Abuse	
Drugmaker		Generic	
Charges		Suffering	
Treat		Drop	
Suboxone		Drugs	
Widely			
Share			
Stock			
Used			
Exchange			
Suffering			
Opioid			
Addiction			

Table 4.4: Comparing predicted words over the 4 methods.

4.4.2 Results for Experiment 2

What difference we observe mainly here is the difference in attention weights for the words. While the ones with lower weights are not very consequential, the words with greater attention weights change when the embedding dimensions are varied. Also, the models overfit within 6-7 epochs as can be seen in the Figure 3.3. As mentioned at the start of this section, one must look beyond how trust-able is our interpretable model. We need to evaluate as humans to believe that our model is right in predicting that particular class. The Tables 4.5-4.8 below compare the words predicted by the attention network and the respective interpretable models. The color coding is as follows: Yellow - 2 similar words ; Light Green - 3 similar words ; Dark Green - All models have the words

Attention words for Embedding Dimension				LIME	SHAP	ELI5
50	100	200	300			
Opioid	News	Exchange	Company	Cheaper	Opioid	Opioid
Stock	Department	Addiction	Department	Opioid	Prosecutors	Cheaper
Share	Company	London	News	Stock	Safer	Stock
Plunged	Justice	Opioid	Opioid	Company	Plunged	Exchange
Treat	Indicted	Stock	Stock	Exchange	Indicted	Generic
People	Addiction	Justice	Share	Generic	Falsely	Prices
Company	Make	Suffering	Plunged	Prices	Cheaper	Treat
Department	British	Department	Treat	Treat	Share	Fraud
News	Drug	Plunged	People	Share	Prone	Company
Justice	Plunged	Treat	Conspiracy	Marketed	Fraud	Share
Indicted	Conspiracy	People	Fraud	Fraud	Conspiracy	Federal
Conspiracy	Fraud	News	Today	Radio	Cheaper	Marketed
Fraud	Today	Drug	Price		Addiction	Drop
People	People	Share	Plunged		Treat	Today
London	London	Company	Drugmaker		Charges	Radio
Drugmaker	Price	Price	Charges		Suffering	Public
British	Drugmaker	Drop	British		Abuse	Department
	Charges	Conspiracy	Indicted		Drugs	
	Treat	Fraud	Addiction		Generic	
	Suboxone	Indicted	Came		Drop	
	Widely	Today	Make			
	Share	British	Widely			
	Stock	British	Suffering			
	Exchange	Drugmaker	Drop			
	Suffering	Conspiracy	Justice			
	Opioid	Make	Suboxone			
			Indivior			

Table 4.5: Comparing the methods for Embedding Dimension - 50

Attention words for Embedding Dimension				LIME	SHAP	ELI5
50	100	200	300			
Opioid	News	Exchange	Company	Cheaper	Opioid	Opioid
Stock	Department	Addiction	Department	Opioid	Prosecutors	Cheaper
Share	Company	London	News	Stock	Safer	Stock
Plunged	Justice	Opioid	Opioid	Company	Plunged	Exchange
Treat	Indicted	Stock	Stock	Exchange	Indicted	Generic
People	Addiction	Justice	Share	Generic	Falsely	Prices
Company	Make	Suffering	Plunged	Prices	Cheaper	Treat
Department	British	Department	Treat	Treat	Share	Fraud
News	Drug	Plunged	People	Share	Prone	Company
Justice	Plunged	Treat	Conspiracy	Marketed	Fraud	Share
Indicted	Conspiracy	People	Fraud	Fraud	Conspiracy	Federal
Conspiracy	Fraud	News	Today	Radio	Cheaper	Marketed
Fraud	Today	Drug	Price		Addiction	Drop
People	People	Share	Plunged		Treat	Today
London	London	Company	Drugmaker		Charges	Radio
Drugmaker	Price	Price	Charges		Suffering	Public
British	Drugmaker	Drop	British		Abuse	Department
	Charges	Conspiracy	Indicted		Drugs	
	Treat	Fraud	Addiction		Generic	
	Suboxone	Indicted	Came		Drop	
	Widely	Today	Make			
	Share	British	Widely			
	Stock	British	Suffering			
	Exchange	Drugmaker	Drop			
	Suffering	Conspiracy	Justice			
	Opioid	Make	Suboxone			
			Indivior			

Table 4.6: Comparing the methods for Embedding Dimension - 100

Attention words for Embedding Dimension				LIME	SHAP	ELI5
50	**100**	**200**	**300**			
Opioid	News	Exchange	Company	Cheaper	Opioid	Opioid
Stock	Department	Addiction	Department	Opioid	Prosecutors	Cheaper
Share	Company	London	News	Stock	Safer	Stock
Plunged	Justice	Opioid	Opioid	Company	Plunged	Exchange
Treat	Indicted	Stock	Stock	Exchange	Indicted	Generic
People	Addiction	Justice	Share	Generic	Falsely	Prices
Company	Make	Suffering	Plunged	Prices	Cheaper	Treat
Department	British	Department	Treat	Treat	Share	Fraud
News	Drug	Plunged	People	Share	Prone	Company
Justice	Plunged	Treat	Conspiracy	Marketed	Fraud	Share
Indicted	Conspiracy	People	Fraud	Fraud	Conspiracy	Federal
Conspiracy	Fraud	News	Today	Radio	Cheaper	Marketed
Fraud	Today	Drug	Price		Addiction	Drop
People	People	Share	Plunged		Treat	Today
London	London	Company	Drugmaker		Charges	Radio
Drugmaker	Price	Price	Charges		Suffering	Public
British	Drugmaker	Drop	British		Abuse	Department
	Charges	Conspiracy	Indicted		Drugs	
	Treat	Fraud	Addiction		Generic	
	Suboxone	Indicted	Generic		Drop	
	Widely	Today	Make			
	Share	Cheaper	Widely			
	Stock	British	Suffering			
	Exchange	Drugmaker	Drop			
	Suffering	Came	Justice			
	Opioid	Make	Suboxone			
			Indivior			

Table 4.7: Comparing the methods for Embedding Dimension - 200

Attention words for Embedding Dimension				LIME	SHAP	ELI5
50	100	200	300			
Opioid	News	Exchange	Company	Cheaper	Opioid	Opioid
Stock	Department	Addiction	Department	Opioid	Prosecutors	Cheaper
Share	Company	London	News	Stock	Safer	Stock
Plunged	Justice	Opioid	Opioid	Company	Plunged	Exchange
Treat	Indicted	Stock	Stock	Exchange	Indicted	Generic
People	Addiction	Justice	Share	Generic	Falsely	Prices
Company	Make	Suffering	Plunged	Prices	Cheaper	Treat
Department	British	Department	Treat	Treat	Share	Fraud
News	Drug	Plunged	People	Share	Prone	Company
Justice	Plunged	Treat	Conspiracy	Marketed	Fraud	Share
Indicted	Conspiracy	People	Fraud	Fraud	Conspiracy	Federal
Conspiracy	Fraud	News	Today	Radio	Cheaper	Marketed
Fraud	Today	Drug	Price		Addiction	Drop
People	People	Share	Exchange		Treat	Today
London	London	Company	Drugs		Charges	Radio
Drugmaker	Price	Price	Charges		Suffering	Public
British	Drugmaker	Drop	British		Abuse	Department
	Charges	Conspiracy	Indicted		Drugs	
	Treat	Fraud	Addiction		Generic	
	Suboxone	Indicted	Generic		Drop	
	Widely	Today	Make			
	Share	Cheaper	Widely			
	Stock	British	Suffering			
	Exchange	Drugmaker	Drop			
	Suffering	Came	Justice			
	Opioid	Make	Suboxone			
			Indivior			

Table 4.8: Comparing the methods for Embedding Dimension - 300

It is very evidently seen from these tables that as we increase the dimension of the embedding, there is a greater overlap between the words from the attention network and the interpretable models. Particularly, ELI5 has a built in evaluation metric - “score" which is an accuracy score weighted by cosine distance between generated sample and the original document (i.e. texts which are closer to the example are more important). Accuracy shows how good are ‘top 1’ predictions. ELI5 also tells the mean KL divergence for all target classes. The values are listed below for each model.

Embedding Dimension	Score	Mean KL-Divergence
50	0.9398	0.0786
100	0.9403	0.0569
200	0.9397	0.0566
300	0.9410	0.0566

Table 4.9: Score and KL-Divergence values for ELI5

4.4.3 Result for Experiment 3

XAI can be used to tune model hyper-parameters. In the following example, I will show a case where LIME is used to generate explanations for a particular test example of 300 dimension embedding as it trains.

Figure 4.7: Explainability per Epoch - MTC dataset

4.4.4 Result for Experiment 4 - Application

We have the following test example.

Input text: Catheterization laboratory events and hospital outcome with direct angioplasty for acute myocardial infarction To assess the safety of direct infarct angioplasty without antecedent thrombolytic therapy, catheterization laboratory and hospital events were assessed in consecutively treated patients with infarctions involving the left anterior descending (n = 100 patients), right (n = 100), and circumflex (n = 50) coronary arteries. The groups of patients were similar for age (left anterior descending coronary artery, 59 years; right coronary artery, 58 years; circumflex coronary artery, 62 years), patients with multivessel disease (left anterior descending coronary artery, 55%; right coronary artery, 55%; circumflex coronary artery, 64%), and patients with initial grade 0/1 antegrade flow (left anterior descending coronary artery, 79%; right coronary artery, 84%; circumflex coronary artery, 90%). Cardiogenic shock was present in eight patients with infarction of the left anterior descending coronary artery, four with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery. Major catheterization laboratory events (cardioversion, cardiopulmonary resuscitation, dopamine or intra-aortic balloon pump support for hypotension, and urgent surgery) occurred in 10 patients with infarction of the left anterior descending coronary artery, eight with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery (16 of 16 shock and six of 234 nonshock patients, p less than 0.001). There was one in-laboratory death (shock patient with infarction of the left anterior descending coronary artery).

Output topic from Topic Classifier: Cardiovascular Diseases

The figures 4.8 - 4.10 below show the outputs from each XAI technique discussed and the tables 4.10-4.13 that follow compare the embedding dimension variants .

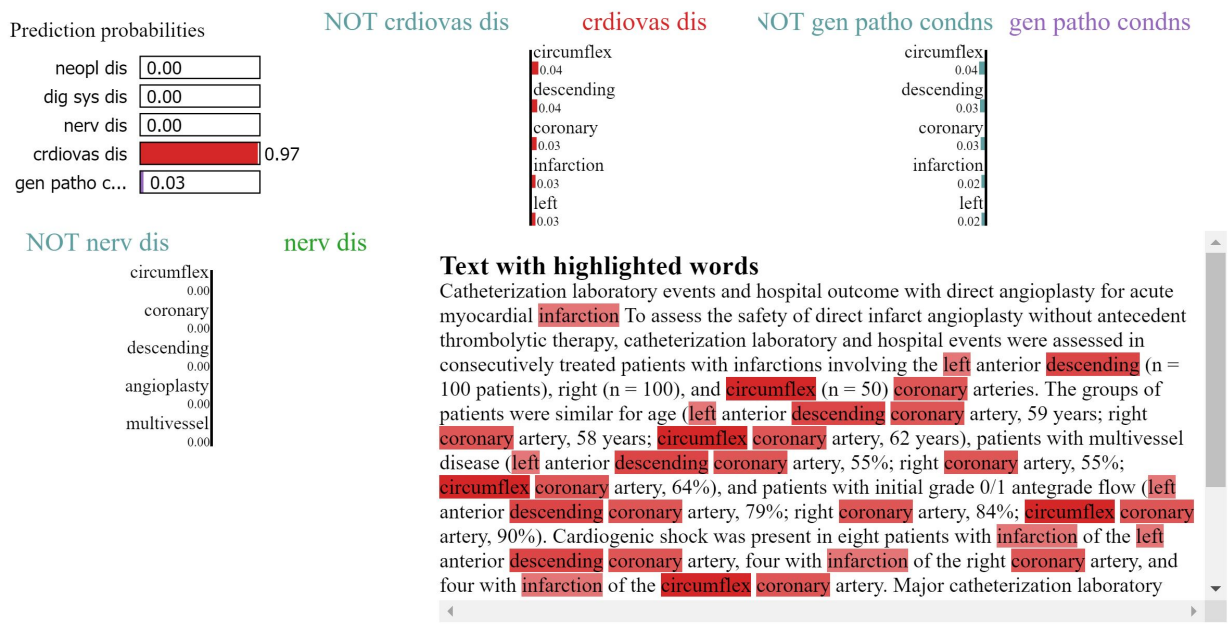


Figure 4.8: MTC - Output from LIME

y=neoplastic diseases (probability **0.000**, score **-40.602**) top features

Contribution[?]	Feature
-0.103	<BIAS>
-40.499	Highlighted in text (sum)

catheterization laboratory events and hospital outcome with direct angioplasty for acute myocardial infarction to assess the safety of direct infarct angioplasty without antecedent thrombolytic therapy, catheterization laboratory and hospital events were assessed in consecutively treated patients with infarctions involving the left anterior descending (n = 100 patients), right (n = 100), and circumflex (n = 50) coronary arteries. the groups of patients were similar for age (left anterior descending coronary artery, 59 years; right coronary artery, 58 years; circumflex coronary artery, 62 years), patients with multivessel disease (left anterior descending coronary artery, 55%; right coronary artery, 55%; circumflex coronary artery, 64%), and patients with initial grade 0/1 antegrade flow (left anterior descending coronary artery, 79%; right coronary artery, 84%; circumflex coronary artery, 90%). cardiogenic shock was present in eight patients with infarction of the left anterior descending coronary artery, four with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery. major catheterization laboratory events (cardioversion, cardiopulmonary resuscitation, dopamine or intra-aortic balloon pump support for hypotension, and urgent surgery) occurred in 10 patients with infarction of the left anterior descending coronary artery, eight with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery (16 of 16 shock and six of 234 nonshock patients, p less than 0.001). there was one in-laboratory death (shock patient with infarction of the left anterior descending coronary artery).

y=digestive system diseases (probability **0.000**, score **-38.148**) top features

Contribution[?]	Feature
-0.149	<BIAS>
-37.999	Highlighted in text (sum)

catheterization laboratory events and hospital outcome with direct angioplasty for acute myocardial infarction to assess the safety of direct infarct angioplasty without antecedent thrombolytic therapy, catheterization laboratory and hospital events were assessed in consecutively treated patients with infarctions involving the left anterior descending (n = 100 patients), right (n = 100), and circumflex (n = 50) coronary arteries. the groups of patients were similar for age (left anterior descending coronary artery, 59 years; right coronary artery, 58 years; circumflex coronary artery, 62 years), patients with multivessel disease (left anterior descending coronary artery, 55%; right coronary artery, 55%; circumflex coronary artery, 64%), and patients with initial grade 0/1 antegrade flow (left anterior descending coronary artery, 79%; right coronary artery, 84%; circumflex coronary artery, 90%). cardiogenic shock was present in eight patients with infarction of the left anterior descending coronary artery, four with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery. major catheterization laboratory events (cardioversion, cardiopulmonary resuscitation, dopamine or intra-aortic balloon pump support for hypotension, and urgent surgery) occurred in 10 patients with infarction of the left anterior descending coronary artery, eight with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery (16 of 16 shock and six of 234 nonshock patients, p less than 0.001). there was one in-laboratory death (shock patient with infarction of the left anterior descending coronary artery).

Figure 4.9: MTC - Output from ELI5

y=nervous system diseases (probability **0.000**, score **-18.604**) top features

Contribution[?]	Feature
-0.162	<BIAS>
-18.442	Highlighted in text (sum)

catheterization laboratory events and hospital outcome with direct angioplasty for acute myocardial infarction to assess the safety of direct infarct angioplasty without antecedent thrombolytic therapy, catheterization laboratory and hospital events were assessed in consecutively treated patients with infarctions involving the left anterior descending (n = 100 patients), right (n = 100), and circumflex (n = 50) coronary arteries. the groups of patients were similar for age (left anterior descending coronary artery, 59 years; right coronary artery, 58 years; circumflex coronary artery, 62 years), patients with multivessel disease (left anterior descending coronary artery, 55%; right coronary artery, 55%; circumflex coronary artery, 64%), and patients with initial grade 0/1 antegrade flow (left anterior descending coronary artery, 79%; right coronary artery, 84%; circumflex coronary artery, 90%). cardiogenic shock was present in eight patients with infarction of the left anterior descending coronary artery, four with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery. major catheterization laboratory events (cardioversion, cardiopulmonary resuscitation, dopamine or intra-aortic balloon pump support for hypotension, and urgent surgery) occurred in 10 patients with infarction of the left anterior descending coronary artery, eight with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery (16 of 16 shock and six of 234 nonshock patients, p less than 0.001). there was one in-laboratory death (shock patient with infarction of the left anterior descending coronary artery).

y=cardiovascular diseases (probability **0.907**, score **4.058**) top features

Contribution[?]	Feature
+3.970	Highlighted in text (sum)
+0.088	<BIAS>

catheterization laboratory events and hospital outcome with direct angioplasty for acute myocardial infarction to assess the safety of direct infarct angioplasty without antecedent thrombolytic therapy, catheterization laboratory and hospital events were assessed in consecutively treated patients with infarctions involving the left anterior descending (n = 100 patients), right (n = 100), and circumflex (n = 50) coronary arteries. the groups of patients were similar for age (left anterior descending coronary artery, 59 years; right coronary artery, 58 years; circumflex coronary artery, 62 years), patients with multivessel disease (left anterior descending coronary artery, 55%; right coronary artery, 55%; circumflex coronary artery, 64%), and patients with initial grade 0/1 antegrade flow (left anterior descending coronary artery, 79%; right coronary artery, 84%; circumflex coronary artery, 90%). cardiogenic shock was present in eight patients with infarction of the left anterior descending coronary artery, four with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery. major catheterization laboratory events (cardioversion, cardiopulmonary resuscitation, dopamine or intra-aortic balloon pump support for hypotension, and urgent surgery) occurred in 10 patients with infarction of the left anterior descending coronary artery, eight with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery (16 of 16 shock and six of 234 nonshock patients, p less than 0.001). there was one in-laboratory death (shock patient with infarction of the left anterior descending coronary artery).

y=general pathological conditions (probability **0.093**, score **-2.183**) top features

Contribution[?]	Feature
-0.155	<BIAS>
-2.028	Highlighted in text (sum)

catheterization laboratory events and hospital outcome with direct angioplasty for acute myocardial infarction to assess the safety of direct infarct angioplasty without antecedent thrombolytic therapy, catheterization laboratory and hospital events were assessed in consecutively treated patients with infarctions involving the left anterior descending (n = 100 patients), right (n = 100), and circumflex (n = 50) coronary arteries. the groups of patients were similar for age (left anterior descending coronary artery, 59 years; right coronary artery, 58 years; circumflex coronary artery, 62 years), patients with multivessel disease (left anterior descending coronary artery, 55%; right coronary artery, 55%; circumflex coronary artery, 64%), and patients with initial grade 0/1 antegrade flow (left anterior descending coronary artery, 79%; right coronary artery, 84%; circumflex coronary artery, 90%). cardiogenic shock was present in eight patients with infarction of the left anterior descending coronary artery, four with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery. major catheterization laboratory events (cardioversion, cardiopulmonary resuscitation, dopamine or intra-aortic balloon pump support for hypotension, and urgent surgery) occurred in 10 patients with infarction of the left anterior descending coronary artery, eight with infarction of the right coronary artery, and four with infarction of the circumflex coronary artery (16 of 16 shock and six of 234 nonshock patients, p less than 0.001). there was one in-laboratory death (shock patient with infarction of the left anterior descending coronary artery).

Figure 4.9: MTC - Output from ELI5 (continued)

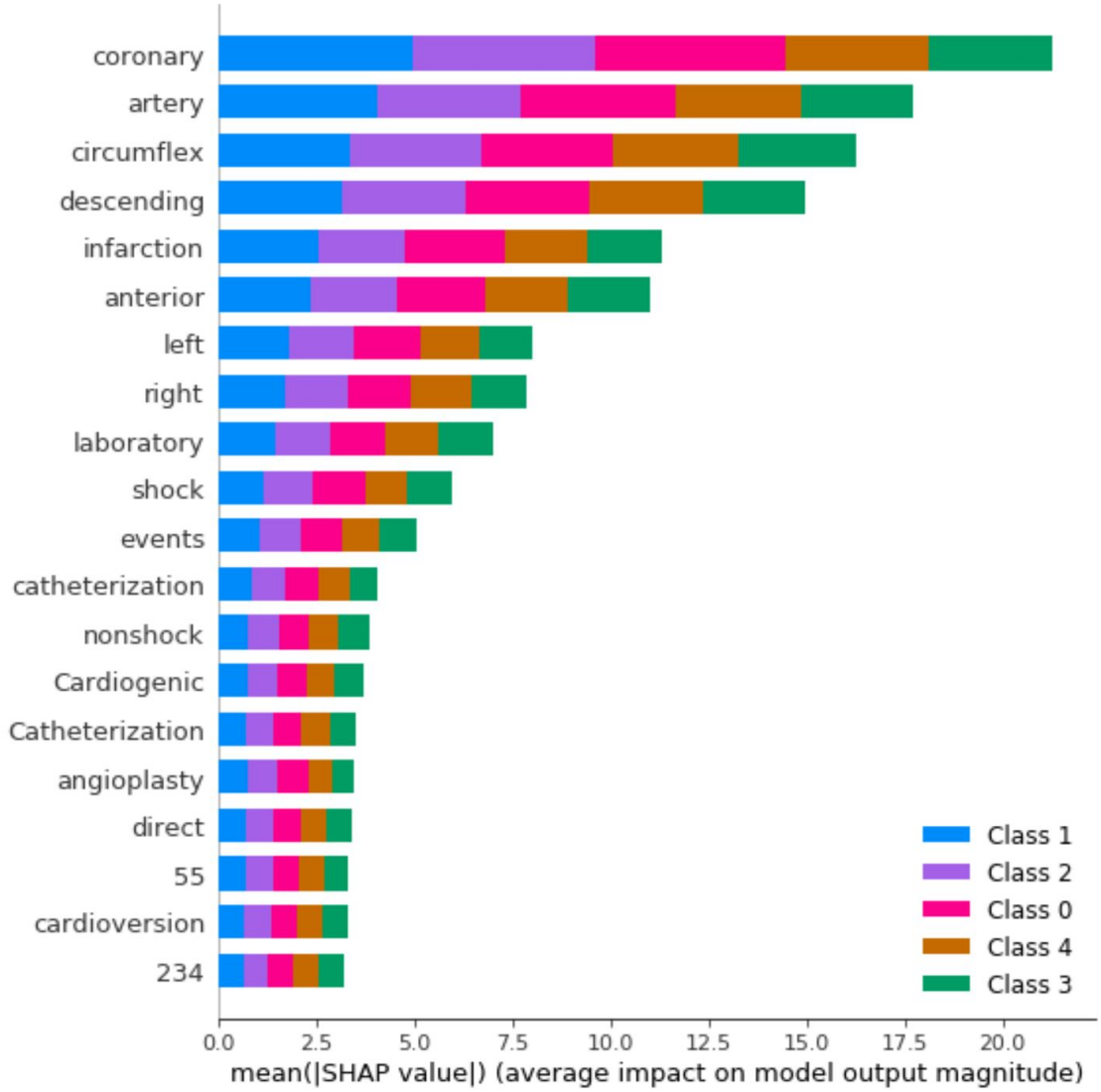


Figure 4.10: MTC - SHAP summary plot - Class 0 - Neoplastic Diseases, Class 1 -Digestive System Diseases, Class 2 -Nervous System Diseases, Class 3 -Cardiovascular Diseases and Class 5 -General Pathological Conditions.

Attention words for Embedding Dimension				LIME	SHAP	ELI5
50	100	200	300			
occurred	infarction	angioplasty	artery	infarction	coronary	laboratory
myocardial	myocardial	thrombolytic	eight	left	artery	acute
catheterization	acute	descending	right	descending	circumflex	infarction
coronary	angioplasty	cardiogenic	descending	circumflex	infarction	myocardial
infarction	left	infarction	thrombolytic	coronary	anterior	coronary
cardioversion	hypotension	artery	patient	catheterization	left	artery
acute	circumflex	myocardial	myocardial	angioplasty	right	cardioversion
left	right	coronary	angioplasty	infarct	laboratory	circumflex
anterior	coronary	anterior	coronary	myocardial	shock	shock
angioplasty	cardiogenic	antecedent	present	artery	events	left
artery	anterior	shock	involving		catheterization	cardiopulmonary
patient	cardiopulmonary	left	acute		cardiogenic	anterior
hypotension	artery	acute	infarction		angioplasty	
thrombolytic	thrombolytic	circumflex	anterior		cardioversion	
circumflex	shock	multivessel	circumflex			
laboratory	catheterization	disease	left			
cardiopulmonary	cardioversion	patient	antegrade			
antegrade	antecedent	therapy	multivessel			
surgery	dopamine	antegrade	shock			
antecedent	laboratory	catheterization	cardiogenic			
	hospital	safety	infarct			
	initial	laboratory	catheterization			
			safety			
			laboratory			

Table 4.10: MTC - Comparing the methods for Embedding Dimension - 50

Attention words for Embedding Dimension				LIME	SHAP	ELI5
50	100	200	300			
occurred	infarction	angioplasty	artery	infarction	coronary	laboratory
myocardial	myocardial	thrombolytic	eight	left	artery	acute
catheterization	acute	descending	right	descending	circumflex	infarction
coronary	angioplasty	cardiogenic	descending	circumflex	infarction	myocardial
infarction	left	infarction	thrombolytic	coronary	anterior	coronary
cardioversion	hypotension	artery	patient	catheterization	left	artery
acute	circumflex	myocardial	myocardial	angioplasty	right	cardioversion
left	right	coronary	angioplasty	infarct	laboratory	circumflex
anterior	coronary	anterior	coronary	myocardial	shock	shock
angioplasty	cardiogenic	antecedent	present	artery	events	left
artery	anterior	shock	involving	anterior	catheterization	cardiopulmonary
patient	cardiopulmonary	left	acute		cardiogenic	anterior
hypotension	artery	acute	infarction		angioplasty	catherterization
thrombolytic	thrombolytic	circumflex	anterior		cardioversion	
circumflex	shock	multivessel	circumflex			
laboratory	catheterization	disease	left			
cardiopulmonary	cardioversion	patient	antegrade			
antegrade	antecedent	therapy	multivessel			
surgery	dopamine	antegrade	shock			
antecedent	laboratory	catheterization	cardiogenic			
	hospital	safety	infarct			
	initial	laboratory	catheterization			
			safety			
			laboratory			

Table 4.11: MTC - Comparing the methods for Embedding Dimension - 100

Attention words for Embedding Dimension				LIME	SHAP	ELI5
50	100	200	300			
occurred	infarction	angioplasty	artery	infarction	coronary	laboratory
myocardial	myocardial	thrombolytic	eight	left	artery	acute
catheterization	acute	descending	right	descending	circumflex	infarction
coronary	angioplasty	cardiogenic	descending	circumflex	infarction	myocardial
infarction	left	infarction	thrombolytic	coronary	anterior	coronary
cardioversion	hypotension	artery	patient	catheterization	left	artery
acute	circumflex	myocardial	myocardial	angioplasty	right	cardioversion
left	right	coronary	angioplasty	cardiogenic	laboratory	circumflex
anterior	coronary	anterior	coronary	myocardial	shock	shock
angioplasty	cardiogenic	antecedent	present	artery	events	left
artery	anterior	shock	involving	anterior	catheterization	cardiopulmonary
patient	cardiopulmonary	left	acute		cardiogenic	anterior
hypotension	artery	acute	infarction		angioplasty	catherterization
thrombolytic	thrombolytic	circumflex	anterior		cardioversion	
circumflex	shock	multivessel	circumflex			
laboratory	catheterization	disease	left			
cardiopulmonary	cardioversion	patient	antegrade			
antegrade	antecedent	therapy	multivessel			
surgery	dopamine	antegrade	shock			
antecedent	laboratory	catheterization	cardiogenic			
	hospital	safety	infarct			
	initial	laboratory	catheterization			
			safety			
			laboratory			

Table 4.12: MTC - Comparing the methods for Embedding Dimension - 200

Attention words for Embedding Dimension				LIME	SHAP	ELI5
50	100	200	300			
occurred	infarction	angioplasty	artery	infarction	coronary	laboratory
myocardial	myocardial	thrombolytic	eight	left	artery	acute
catheterization	acute	descending	right	descending	circumflex	infarction
coronary	angioplasty	cardiogenic	descending	circumflex	infarction	myocardial
infarction	left	infarction	thrombolytic	coronary	anterior	coronary
cardioversion	hypotension	artery	patient	catheterization	left	artery
acute	circumflex	myocardial	myocardial	angioplasty	right	thrombolytic
left	right	coronary	angioplasty	cardiogenic	laboratory	circumflex
anterior	coronary	anterior	coronary	myocardial	shock	shock
angioplasty	cardiogenic	antecedent	present	artery	events	left
artery	anterior	shock	involving	anterior	catheterization	cardiopulmonary
patient	cardiopulmonary	left	acute	laboratory	cardiogenic	anterior
hypotension	artery	acute	infarction		angioplasty	catherterization
thrombolytic	thrombolytic	circumflex	anterior		cardioversion	angioplasty
circumflex	shock	multivessel	circumflex			
laboratory	catheterization	disease	left			
cardiopulmonary	cardioversion	patient	antegrade			
antegrade	antecedent	therapy	multivessel			
surgery	dopamine	antegrade	shock			
antecedent	laboratory	catheterization	cardiogenic			
	hospital	safety	infarct			
	initial	laboratory	catheterization			
			safety			
			laboratory			

Table 4.13: MTC - Comparing the methods for Embedding Dimension - 300

It is very evidently seen from these tables that as we increase the dimension of the embedding, there is a greater overlap between the words from the attention network and the interpretable models.

5. CONCLUSIONS

Semantic vector space models of language represent each word with a real-valued vector. These vectors can be used as features in a variety of applications, such as information retrieval , document classification , question answering , named entity recognition, and parsing. Traditionally, keyword research involved building a list or database of relevant keywords that we hoped to rank for. Often graded by difficulty score, click through rate and search volume, keyword research was about finding candidates in this list to create content and gather some organic traffic through exact matching.

This thesis involved 3 stages that were interconnected. In the first section, we understood how paragraph vectors worked and how to obtain documents similar to a search query based on cosine similarity as a metric. I then analyzed how noise affects the model through negative sampling as well as external Gaussian perturbations.2.10 and 2.11 were self explanatory in terms of how the model behaved.

In Chapter 3, my aim was to improve the document prediction accuracy of documents we saw in Chapter 2. I used a topic classifier to bridge this gap in improving prediction accuracy. In order to understand the working of the topic classifier, I employed a Hierarchical Attention Network as well as an ensemble of shallow learning models (refer 3.3). In order to find the optimal topic classifier, I varied the embedding dimension and compared the performance of the models - Figures 3.5 to 3.8. Once I had the optimal classifier, I employed it to obtain the document based on a search query. I tested this on multiple datasets and the results showed a significant improvement in utilizing this technique. In addition to these, I tested this model on 3 separate datasets to show the use of this in Medical Field, Genetics and Abstract Classification for topics in Science.

In the final Chapter, I justified the actions I took in Chapter 3 using XAI techniques. The methods

listed in the chapter are model agnostic. This means that they cannot peek into the model, but find a local explanation a particular example. LIME, SHAP and ELI5 proved to be extremely useful in showing the consistency of the Topic Classifier. I was also able to prove that the overlap between the words increased with increase in embedding dimension.

REFERENCES

[1] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992.

[2] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[3] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[4] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[5] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[6] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196, 2014.

[7] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[8] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[9] Rob Koopman, Shenghui Wang, and Gwenn Englebienne. Fast and discriminative semantic embedding. In *Proceedings of the 13th International Conference on Computational Semantics-Long Papers*, pages 235–246, 2019.

[10] Dimitris Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of computer and System Sciences*, 66(4):671–687, 2003.

[11] William B Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

[12] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077. International World Wide Web Conferences Steering Committee, 2015.

[13] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems*, pages 585–591, 2002.

[14] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[15] Trevor F Cox, MA Cox, and TF COX. Multidimensional scaling, st, 1994.

[16] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[17] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017.

[18] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196, 2014.

[19] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.

[20] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[21] Ye Zhang and Byron Wallace. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*, 2015.

[22] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 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*, pages 1480–1489, 2016.

[23] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[24] Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. Hdltex: Hierarchical deep learning for text classification. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 364–371. IEEE, 2017.

[25] Benjamin Letham, Cynthia Rudin, Tyler H McCormick, David Madigan, et al. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3):1350–1371, 2015.

[26] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1721–1730. ACM, 2015.

[27] Attend Show. Tell: Neural image caption generation with visual attention. *Kelvin Xu et. al.. arXiv Pre-Print*, 23, 2015.

[28] Goku Mohandas. Interpretability via attentional and memory-based interfaces, using tensorflow, 2017.

[29] Sofia Serrano and Noah A Smith. Is attention interpretable? *arXiv preprint arXiv:1906.03731*, 2019.

[30] Sarthak Jain and Byron C Wallace. Attention is not explanation. *arXiv preprint arXiv:1902.10186*, 2019.

[31] Chengliang Yang, Anand Rangarajan, and Sanjay Ranka. Global model interpretation via recursive partitioning. In *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pages 1563–1570. IEEE, 2018.

[32] Marco A Valenzuela-Escárcega, Ajay Nagesh, and Mihai Surdeanu. Lightly-supervised representation learning with global interpretability. *arXiv preprint arXiv:1805.11545*, 2018.

[33] Dumitru Erhan, Aaron Courville, and Yoshua Bengio. Understanding representations learned in deep architectures. *Department dInformatique et Recherche Operationnelle, University of Montreal, QC, Canada, Tech. Rep*, 1355:1, 2010.

[34] Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3429–3437, 2017.

[35] Pieter-Jan Kindermans, Kristof T Schütt, Maximilian Alber, Klaus-Robert Müller, Dumitru Erhan, Been Kim, and Sven Dähne. Learning how to explain neural networks: Patternnet and patternattribution. *arXiv preprint arXiv:1705.05598*, 2017.

[36] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM, 2016.

[37] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. 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 4765–4774. Curran Associates, Inc., 2017.

[38] TeamHG-Memex. Eli5. https://eli5.readthedocs.io/en/latest/overview.html#features.

[39] Donald P Green and Holger L Kern. Modeling heterogeneous treatment effects in large-scale experiments using bayesian additive regression trees. In *The annual summer meeting of the society of political methodology*, 2010.

[40] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 24(1):44–65, 2015.

[41] Robert Andrews, Joachim Diederich, and Alan B Tickle. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-based systems*, 8(6):373–389, 1995.

[42] Paulo Cortez and Mark J Embrechts. Using sensitivity analysis and visualization techniques to open black box data mining models. *Information Sciences*, 225:1–17, 2013.

[43] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[44] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. Model class reliance: Variable importance measures for any machine learning model class, from the "rashomon" perspective. *arXiv preprint arXiv:1801.01489*, 2018.

[45] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in Neural Information Processing Systems*, pages 2280–2288, 2016.