UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Sander Tars

Multi-Domain Neural Machine Translation

Master's Thesis (30 ECTS)

Supervisor: Mark Fišel, PhD

Tartu 2018

Multi-Domain Neural Machine Translation

Abstract: In this thesis we present an approach to neural machine translation (NMT) that supports multiple domains in a single model and allows switching between the domains when translating. The core idea is to treat text domains as distinct languages and use multilingual NMT methods to create multi-domain translation systems; we show that this approach results in significant translation quality gains over fine-tuning. We also propose approach of unsupervised domain assignment and explore whether the knowledge of pre-specified text domains is necessary; turns out that it is after all, but also that when it is not known quite high translation quality can be reached, and even higher than with known domains in some cases. Additionally, we explore the possibility of intra-language style adaptation through zero shot translation. We show that this approach is able to style adapt, however, with unresolved text deterioration issues.

Keywords: Neural machine translation, unsupervised clustering, zero shot translation, multi-domain, domain tuning, style adaptation

CERCS: P176 Artificial intelligence

Mitme-domeenne tehisnärvivõrkudel põhinev masintõlge

Lühikokkuvõte: Käesolev magistritöö kätkeb endas neurotõlke lähenemist, mis toetab mitme-domeenseid tekste ja võimaldab tõlkimisel arvestada domeenide eripära. Antud lähenemine lähtub põhimõttest, et me käsitleme domeene kui eraldiseisvaid keeli, ning kasutame nende tõlkimiseks mitmekeelse neurotõlke meetodeid. Samuti näitame et mainitud lähenemise tulemusena tõlkekvaliteedi hinnang paraneb märgatavalt. Käesolevas töös pakume välja ka lähenemise domeenide automaatseks määramiseks ja uurime kas eelnev domeenijaotuse info on üldse vajalik. Tuleb välja, et on, kuid kui sellist infot ei ole, on automaatset määramist kasutades võimalik samuti kõrge tõlkekvaliteedini jõuda, kohati isegi kõrgemani kui eelnevat domeenijaotuse infot kasutades. Lisaks uurime selles töös, kas keelesisene stiilile kohandamine tühipauk(*zero-shot*) tõlke kaudu on võimalik. Näitame, et see lähenemine on võimeline stiilile kohanduma, kuid koos siiani lahenduseta kvaliteedilangusega.

Võtmesõnad: neurotõlge, juhendamata klasterdamine, tühipauk(*zero-shot*) tõlge, mitme-domeenne, domeenile sobitamine, stiili kohandamine

CERCS: P176 Tehisintellekt

Contents

1 Introduction

It the last decade, machine translation (MT) has become widely used tool. The uses range from translating manuals and professional texts to translating texts in order to create parallel training data for MT itself. There are several reasons for the advance of MT.

Firstly, the amount of parallel data has increased for many languages to the level, where it is sufficient to train usable MT systems. There are many parallel data sources, for example OpenSubtitles (Lison and Tiedemann, 2016) and European parliament texts (Koehn, 2005).

Secondly, the increase in computational power and use of GPU-s has allowed the use of artificial neural networks. Neural networks have been shown to produce results superior to statistical machine translation when enough data is provided. For this reason, many MT systems, for example Google Translate, have migrated to neural machine translation (NMT) and MT has become almost synonymous with NMT.

The field of NMT is still rapidly evolving and there are many aspects to be improved. One of these is the domain effect on translation quality. Data-driven machine translation systems, like NMT systems, depend on the text domain of their training data. In a typical in-domain MT scenario the amount of parallel texts from a single domain is not enough to train a good translation system, even more so for neural machine translation (NMT, Bahdanau et al., 2014); thus models are commonly trained on a mixture of parallel texts from different domains and then later trained on in-domain texts to fine-tune the model to specific domain (Luong and Manning, 2015).

In-domain fine-tuning has two main shortcomings: it depends on the availability of sufficient amounts of in-domain data in order to avoid overfitting and it results in degraded performance for all other domains. The latter means that for translating multiple domains one has to run an individual NMT system for each domain.

In this thesis we treat text domains as distinct languages: for example, instead of English-to-Estonian translation is seen as translating English news to Estonian news. We use the idea of viewing domain as a distinct language in all three parts of the thesis.

In the first part of the thesis we test two multilingual NMT approaches (Johnson et al., 2016; Östling and Tiedemann, 2017) in a bilingual multi-domain setting and show that both outperform single-domain fine-tuning on all the text domains in conducted experiments.

However, this only works when the text domain is known both when training and translating. In some cases the text domain of the input segment is unknown – for example, web MT systems have to cope with a variety of text domains. Also, some parallel texts do not have a single domain while they are either a mix of texts from different sources (like crawled corpora) or naturally constitute a highly heterogeneous mix of texts (like subtitles or Wikipedia articles).

In the second part of this thesis, we address these domain knowledge issues by replacing known domains with automatically derived ones. At training time we cluster parallel sentences and then the multi-domain translation approach is applied to these clusters. When translating, we classify the input segments as belonging to one of these clusters and translate with this automatically derived information.

Finally, in the third part of this thesis, we present initial experiments on intra-language translation between different text domains. The aim of these experiments is to explore the possibility of intra-language style adaptation. Since no parallel data is available for such a task we exploit the zero-shot learning feature of Johnson et al. (2016) and train a multilingual multi-domain NMT system.

In the following related work is reviewed in Section 3, the used methodology of multi-domain NMT and sentence clustering is presented in Section 4. After that, the conducted experiments are described in Section 5, Section 6 and Section 7 and the results are discussed in Section 8 concluding the thesis.

2 Background

In this section the descriptions of most relevant used methods and related definitions is given. These are grouped into four groups: data, vector representations, machine learning, and neural networks.

2.1 Data

Text corpus

There are mainly two types of corpora used in MT - parallel corpora and monolingual corpora.

Parallel language corpus is a set of sentences that are aligned as each others translations. Parallel corpora can be either bilingual or multilingual. Parallel corpora are usually domain-specific. For example, OpenSubtitles corpus concentrates on movie subtitles, EuroParl corpus on the texts of european parliament, and so on. The corpora have different degrees of cleanness, which means how much there are non-parallel sentences.

For low-resource parallel corpus languages, the parallel corpora can be extended through the process of backtranslation. Backtranslation is the process of translating monolingual corpus of target language to source language. The direction is important, so that the text that model learns to generate would be still correct. Backtranslated texts are very noisy, which means they have to be thoroughly cleaned, which is usually done by using attention info of the neural neural network that was used to backtranslate the texts.

Parallel corpus

A **parallel corpus** is a corpus that consists of parallel sentences. In most cases, parallel corpora are bilingual, meaning each parallel sentence pair is between language $L1$ and language $L2$. Parallel corpora can also be multilingual, i.e. they can contain more than two languages. This means that not only sentences between $L1$ and $L2$ are there, but also $L1 - L3$, $L3 - L4$, etc.

Depending on the use case, a bilingual, one-to-many, or many-to-many training corpus may be extracted from parallel corpus, depending on whether bilingual, one-to-many, or many-to-many model is to be trained.

Related to parallel corpora are **comparable corpora**. Comparable corpus consists of texts which are similar in content (same genre, topic, etc.), but the content is not parallel. For example, in this thesis we use Wikipedia as a big comparable corpus where articles are aligned by topic, but content is usually not parallel.

2.2 Vector representations

Embedding

An embedding is a mapping from discrete objects like words, to vectors of real numbers. For example, 100-dimensional word-embeddings could include:

$$word1 : (0.01359, 0.00075997, 0.24608, ..., -0.2524, 1.0048, 0.06259)$$
$$word2 : (0.01396, 0.11887, -0.48963, ..., 0.033483, -0.10007, 0.1158)$$

The dimension values in these vectors typically have no deeper meaning. What holds the information, is the overall patterns of location and distance between vectors - the property that machine learning takes advantage of. Neural networks train best on dense vectors, where the vector values together define an object. Since texts and words do not have a natural dense vector representation, embedding functions have to be applied to create that representation.

Embedding vectors can also serve as an output to evaluate similarity of objects, for example Euclidean distance between vectors applied in nearest neighbor method.

In this thesis, we use embedding vectors in neural networks for word representations and as domain embedding vector. We also use sentence vectors in sequence classification in Section 6.

FastText classification

FastText is an open-source library that allows learning text representations and text classifiers (Bojanowski et al., 2016; Joulin et al., 2016).

One thing that makes fastText vector representations special is that it also considers subword units, unlike word2vec which treats word as the smallest unit. FastText considers words as composed by overlapping n-gram characters. For example, "plant" is composed of [pla, plan, plant],[plant,lant,ant], etc. By default fastText considers n-grams at least three characters long.

FastText also produces fast and accurate text classification. The reasons for good results of fastText are that is uses hierarchical tree structure for categories instead of flat structure, which decreases time complexity from linear to logarithmic in respect to number of classes. Also, fastText uses Huffman coding algorithm to account for imbalanced classes. FastText text low dimensional representation vector is produced by summing the word representation vectors that are present in the text.

In this thesis we use fastText in Section 6 to classify new sequences into automatically generated clusters.

Sent2vec

Sent2vec (Pagliardini et al., 2017) produces numerical vector representations for short texts or sentences similarly to fastText. Sent2vec can be thought of as an unsupervised version of FastText for sentences. Sent2vec is able to produce sentence embedding vectors using word vectors and n-gram embeddings and simultaneously train the composition and embedding vectors.

Sent2vec outperforms most state-of-the-art on most benchmark tasks. Most important for this thesis is the fact that sent2vec vectors show excellent performance on sentence similarity tasks.

In this thesis sent2vec is used in Section 6 to produce sentence vectors in an unsupervised manner. The vectors are used to automatically derive text domains and cluster sentences into those.

2.3 Machine Learning

Byte pair encoding

Byte pair encoding (BPE) (Shibata et al., 1999) is a data compression technique that replaces iteratively the most frequent pair of bytes in a sequence with a single, unused byte, e.g. $aaabdaaabac$ becomes $ZabdZabac$ with the replacement of $aa = Z$. In this thesis an adaption of this algorithm is used for word segmentation (Sennrich et al., 2016a). Instead of merging frequent pairs of bytes, most frequent characters or character sequences are joined in the used approach.

In adapted BPE version the following takes place. Firstly, the vocabulary is initialized with character vocabulary, and each word is represented as a character sequence. A special end-of-word symbol is also added. Then all symbol pair are iteratively counted and replaced with the joined version of most frequent pair, e.g. pair $('c','e')$ becomes $'ce'$. Final vocabulary size is the size of the initial vocabulary, plus the number of join operations, which is given as a input to the algorithm. In this thesis the joint BPE is used, which means that the segmentation is learned on all of the input data jointly and the same segmentation is applied to all input languages.

Over- and underfitting

Overfitted model is a model that is fitted to model the training data too well and loses much of its ability to generalise.

Overfitting is caused by model learning the detail patterns and noise in the training data precisely enough that it has an negative impact on the model performance on new data. The reson for this negative impact is that the patterns learned on training data do not exists in data generally and thus it is called overfitting.

Underfitted model is a model that is too general in modelling both training data and new data. Underfitting is easier to discover than overfitting, because it causes bad performance on both training and new datasets, since it does not learn the characteristics for neither.

KMeans clustering

K-means is a unsupervised clustering algorithm. The KMeans clustering is done in four simple steps. The steps in total aim to minimize an objective function that determines the goodness of the clustering. The objective function can be for example average of squared Euclidean distances from the cluster center of each vector in the specific cluster. The algorithm is composed of the following steps:

1. K points, called centroids, are placed into the clustering space.
2. Each object is assigned to the group that has the closest centroid.
3. When all objects have been assigned a specific, the positions of the K centroids are recalculated based on the points in the cluster.
4. Previous two steps are repeated until the centroids no longer move. This process separates the objects into K clusters.

There are several things to be considered in KMeans clustering, for example how the initial centroids are placed and what is the optimal K.

In this thesis KMeans clustering is used in Section 6 to produce automatic domain segmentation through unsupervised sentence vector clustering.

t-SNE

t-distributed stochastic neighbor embedding (t-SNE) is a machine learning algorithm for visualization (van der Maaten and Hinton, 2008). t-SNE works by mapping high-dimensional datapoints into two or three dimensions for visualization purposes. t-SNE mapping works in a way where similarity of datapoint projections indicates with high probability the similarity of objects.

The projections are achieved in two stages. In the first stage, t-SNE constructs a probability distribution over pairs of high-dimensional objects in such a way that similar objects have much higher probability of being picked than dissimilar objects. In the second stage, the low-dimensional projections are further separated by similar probabilistic distribution in a low-dimensional mapping. Also mechanism is applied that helps avoid crowding the points in center. The benefits of the approach include that it is capable of revealing some important global structure such as clusters while retaining the local structure of the data. The algorithm can use various similarity metrics, for example Euclidean distance.

In this thesis t-SNE is used in Section 6 to examine clustering differences through two-component t-SNE projections of sentence embedding vectors.

2.4 Neural Networks

Recurrent neural network

Recurrent neural networks (RNNs) make use of sequential information. These process every element of a sequence, with the output being dependent on the computations of the previous element sequence. RNNs are set to usually look back on only a few steps.

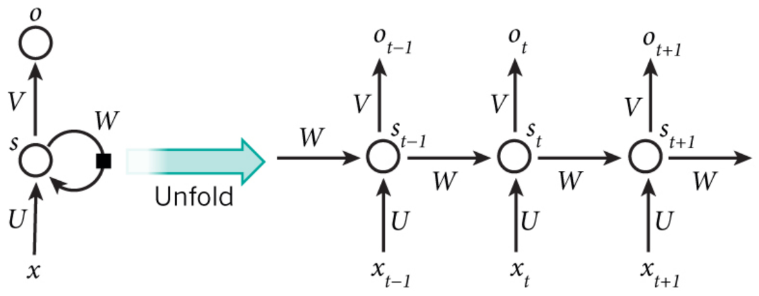

Figure 1. The figure of a typical RNN being unrolled into a full network. Figure is taken from http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

Figure 1 shows a typical RNN which is unrolled into a full network. x_t is the input at time step t. For example, a one-hot vector corresponding to the second word of a sentence. s_t is the hidden state - the "memory" of the network - at time step t. s_t is calculated based on the previous hidden state and the input at the current step. o_t is the output at step t. For example, in predicting the next word in a sentence it would be a vector of probabilities across our vocabulary.

A RNN shares the same parameters (U, V, W on Figure 1) across all steps to reduce the number of parameters learned greatly, compared to regular deep NN-s.

The most commonly used types of RNNs in NLP are LSTMs and GRUs.

LSTM and GRU

Long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) units are parts of LSTM layers of a recurrent neural network (RNN). Such RNN is referred to as an LSTM network. A standard LSTM unit parts are a cell, an input gate, a forget gate, and an output gate. The cell serves as a "memory" by remembering the processed values over some time intervals. The LSTM gates can be thought as regulators for the value that go through it the - hence the denotation "gate". The gates output numbers between zero and one, describing how much of each component should be let through.

LSTMs are well-suited to handle the classification, process and predict time series given time lags of unknown size and duration between important events.

Gated recurrent units (GRU) (Cho et al., 2014) control information flow similarly to LSTM units. However, GRU only has a reset gate and an update gate. These gates control how much each hidden unit remembers while reading or generating a sequence. GRU does not make use of a memory unit. For this reason GRU has fewer parameters than LSTM and thus trains faster.

GRUs and LSTMs perform similarly, thus GRU is often used instead of LSTM because it is more lightweight. The neural network architecture in this thesis also uses GRU.

2.5 Neural Machine Translation

Neural Machine Translation (NMT)

Neural machine translation (NMT) means using the neural network type models to train a statistical model for machine translation. In case of NMT a single system can be trained directly on source and target text to translate, unlike statistical MT that requires the pipeline of systems in between.

The main NMT architecture used is the encoder-decoder architecture with attention mechanism. An RNN encoder-decoder is able to take a sequence as input and generate another sequence as output. Since the simple encoder-decoder architecture has problems with long sequences, an attention mechanism is added to it. The attention mechanism allows the model to learn to place the attention on each entity of input sequence to produce an output sequence.

Interestingly, the encoder-decoder architecture allows for NMT to encode-decode several language pairs within a single model. Modeling several language pairs (e.g Estonian–English and Finnish–Russian) is called multilingual NMT. To model this, both Estonian–English and Finnish–Russian would have to be present in training sentences with appropriate distinction. This multilingual NMT can be achieved in several ways, for example using language tags as in Johnson et al. (2016).

BLEU score

BLEU (Papineni et al., 2002) is a metric for evaluating the quality of machine translation. BLEU is one of the most popular automatic metrics. The popularity of BLEU is due to its good correlation with human judgement, speed, language-independence, and the fact that it is inexpensive to run compared to a human evaluator. BLEU is designed to work at a corpus level as a average scoring of individual sentences, and should not be used for individual sentences. The BLEU scores range from 0 to 1. Score 1 means that the translation sequences and reference sequences are identical – something which rarely happens, thus even human evaluators do not achieve BLEU score 1.

BLEU is calculated as

$$BLEU = BP * exp(\sum_{n=1}^{N} w_n \log p_n), \tag{1}$$

where

$$BP = \begin{cases} 1, & \text{if } c > r. \\ e^{(1-r/c)}, & \text{if } c \leq r. \end{cases} \tag{2}$$

Where w_n in (1) means n-gram weight, p_n means n-gram precision, N means maximum $n - gram$ length, and BP means brevity penalty. BLEU is usually used with parameters $N = 4$ and $w_n = 1/N$ i.e. uniform weights. Brevity penalty (BP) is used to consider differences in translation lengths. BP calculation is shown in (2), where c is the candidate translation and r the reference translation. As we can see, the penalty is only applied when the candidate translation is shorter than the reference translation.

Several modifications of BLEU exist, but these are still similar in principle to the one described here.

Transfer learning

Transfer learning in machine learning generally means that a model trained for one specific task is reused as the basis for a second machine learning task which has got different characteristics from the original task.

The usage of transfer learning is widespread approach in machine learning, especially for image processing, where it can be used to drastically reduce computational cost. A "pure example" of transfer learning in NLP are the "zero-shot" approach and also the language vector sharing that are descirbed in Johnson et al. (2016).

Transfer learning also discussed in this thesis in Sections 5, 6, and 7 to produce a superior multi-domain NMT model.

3 Related Work

In this section we describe the works related to the integral methods of this thesis.

The baseline to which we compare the translation part of this thesis to is fine-tuning NMT systems to a single text domain (Luong and Manning, 2015). In fine-tuning the NMT system is first trained on a mix of parallel texts from different domains and then fine-tuned via continued training on just the in-domain texts. The method shows improved performance on in-domain test data but degrades performance on other domains.

In Sennrich et al. (2016b) the NMT system is parametrized with one additional input feature (politeness), which is included as part of the input sequence, similarly to one of two approaches in this thesis – the domain tag approach. However, their goal is different from ours.

In Kobus et al. (2017) additional word features are used for specifying the text domain together with the same approach as Sennrich et al. (2016b). Although both methods overlap with the first part or this thesis' work (domain features and domain tags), they only test these methods on pre-specified domains, while this thesis includes automatic domain clustering and identification. Also, they use in-domain trained NMT systems as baselines even for small parallel corpora and do experiments with a different NMT architecture. Finally, their results show very modest improvements, while in our case the improvements are much greater.

Other approaches also define a mixture of domains, for example Britz et al. (2017); Chen et al. (2016), however, both define custom NMT methods. This means that the approaches cannot be easily integrated into modern NMT approaches like Transformer (Vaswani et al., 2017), whereas one of our two approaches - domain tagging, is compatible with Transformers and many other modern NMT approaches.

Additionally, the related approaches limit the experiments to the cases where the text domain is known, whereas we explore the performance in automatic domain assignment also.

4 Methodology

In the following section we describe two different approaches to treating text domains as distinct languages and using multi-lingual methods, resulting in multi-domain NMT models. The first approach is inspired by Google's multilingual NMT (Johnson et al., 2016) and the second one by the cross-lingual language models (Östling and Tiedemann, 2017). The mentioned approaches are only applicable in the scenario where domain knowledge exists. Since domain knowledge can often be absent in training and/or translating, we propose approaches of automatic domain segmentation that can be used to extend both of the approaches to any given text corpus without pre-existing domain knowledge. We also propose substituting the pre-defined domain knowledge with automatic domain segmentation to test if automatic domain segmentation can produce better domain segmentation than pre-defined domain knowledge.

4.1 Domain as a Tag

The first approach is based on Johnson et al. (2016). Their method of multilingual translation is based on training the NMT model on data from multiple language pairs, while appending a token specifying the target language to the beginning of the source sequence. No changes to the NMT architecture are required with this approach. They show that the method improves NMT for all languages involved. As an additional benefit this approach causes no increase in the number of parameters, since all language pairs are included in the same model.

We adapt the language tag approach to text domains, appending the domain ID to each source sentence; thus, for instance, "How you doin' ?" from OpenSubtitles2016 (Lison and Tiedemann, 2016) becomes "__OpenSubs How you doin' ?".

The described method has two advantages. Firstly, it is independent of the NMT architecture, and scaling to more domains means simply adding data for these domains. One can assign a domain to each sentence pair of the training set sentence pair, or set the domain to "other" for sentences whose domain we cannot or do not want to identify.

Secondly, they state that in a multilingual NMT model, all parameters are implicitly shared by all the language pairs being modeled. This forces the model to generalize across language boundaries during training. It is observed in the underlying article that when language pairs with little available data and language pairs with abundant data are mixed into a single model, translation quality on the low resource language pair is significantly improved.

We expect this effect to be even more useful for text domains. Traditional tuning to a low-resource domain, or for any specific domain for that matter, would result in a likely over-fitting to that domain. This approach, where all parameters are shared, learns target domain representations without harming other domains' results while maintaining the ability to generalize also on in-domain translation, because little to no over-fitting will be caused. Furthermore, since domains are much more similar than languages, the parameter sharing is expected to have a stronger effect.

4.2 Domain as a Feature

The second approach is based on Östling and Tiedemann (2017) for continuous multilingual language models. The authors propose to use a single RNN model with language vectors that indicate what language is used. As a result each language gets its own embedding, thus ending up with a language model with a predictive distribution $p(x_t|x_{1...t-1}, l)$ which is a continuous function of the language vector l.

In this thesis, we implement the same idea via word features of Nematus (Sennrich et al., 2017), with their learned embeddings replacing the language vector of Östling and Tiedemann (2017). For example, translating "This is a sentence ." to the Estonian

Wikipedia domain would mean an input of "This|2wi is|2wi a|2wi sentence|2wi .|2wi"[1]

Having a single language model learn several languages helps similar languages improve each others representations (Östling and Tiedemann, 2017). Also, they point out that this greatly alleviates the problem of sparse data for smaller languages. We expect the same effect to hold for text domains, especially since similarity between different domains of the same languages is higher than between different languages. Moreover, similarly to the domain tag approach, the usage of many domains in one model helps bypass the over-fitting problem of smaller domains.

4.3 Automatic domain tags

Te aforementioned approaches are only applicable in a scenario where domain knowledge exists. Since for many scenarios there is no pre-existing domain knowledge, we propose the domain of each of the source–target sentence pair to be defined automatically. We take two different approaches to achieve the annotation.

We apply Supervised approach only in single domain setting. It involves assigning categories to roughly 10,000 Wikipedia articles, for which it could be done with high certainty. Assigning categories to more articles is problematic, because the categories assigned in Wikipedia can often be misleading in terms of content. Next each sentence is tagged with the article category.

After tagging the sentences, we train a FastText (Bojanowski et al., 2016; Joulin et al., 2016) classification model with default settings and apply it to classify the rest of the sentences that were not classified based on the article categories. We tag the test/dev set sentences using the same FastText model that is used to cluster training data.

We apply Unupervised approach also to sentence-split data. We treat multi-domain data still as a single domain data of which no domain structure knowledge exists. In this approach we train a sent2vec (Pagliardini et al., 2017) model and use it to calculate sentence vectors in an unsupervised manner. After that, we apply KMeans clustering to identify the clusters in the set of calculated sentence vectors. Finally, we tag each sentence with the label that it was assigned by KMeans. To find the optimal number of clusters, we create several versions with different numbers of clusters.

To tag the test/dev set sentences, we train a FastText supervised classification model on the tagged training set. For each of the cluster versions and for each language pair, we train a separate FastText model. The additional benefit of this kind of clustering is that each new input sentence can be efficiently assigned its cluster. Also, because of more potentially homogenous train-set clusters, the new sentence is hypothetically assigned more appropriate domain than it would be assigned in case of the pre-defined domains.

[1]the "|" is a special symbol in Nematus for delimiting input features.

The potential benefit of the unsupervised approach over supervised approach is that it does not assume any prior knowledge of the data and thus the domain structure does not rely on potentially faulty pre-defined domain structure. This in turn allows the multi-domain translation approach to be applied to any data without the knowledge of its domain structure.

4.4 Zero-shot style adaptation

Another shown feature in Johnson et al. (2016) is that modeling several language pairs in a single model enables translation between language pairs the trained model has never seen in this combination during training. For example, a multilingual NMT model trained with Portuguese→English and English→Spanish examples can generate reasonable translations for Portuguese→Spanish although it has not seen any data for that language pair. This phenomenon is referred to as zero-shot translation. Zero-shot translation serves as an example of transfer learning within NMT models.

We apply this approach to test the possibility of within-language style adaption through zero-shot translation. This would mean that having training examples of $ET\_medicine - EN\_medicine$ and $EN\_news - ET\_news$ could be used to "translate" $ET\_medicine - ET\_news$. Having this kind of style adaptation could potentially be used to simplify scientific text to be readable by wider audiences. Also, this could be used to improve user experience in applications like chatbots, by adapting to the style that client uses, e.g transferring $EN\_general$ to $EN\_manchester$. The reason why we try to use zero-shot translation, is that there is no universal parallel data for direct style adaption through NMT.

To test the possibility of style adaption we train a domain-tagged EN-ET-EN NMT model. We train this model on data that contains both EN–ET and ET–EN parallel sentences. We tag each source sentence with language and domain tag. On the source side for instance, “How you doin’ ?” from OpenSubtitles2016 becomes “__toETOpenSubs How you doin’ ?”. Similarly, on the source side “Kuidas sul läheb ?” from OpenSubtitles2016 becomes “__toENOpenSubs Kuidas sul läheb ?”.

During translating, the Opensubs source sentence “__toETOpenSubs How you doin’ ?” would be given for example a tag “__toENEuroparl” to see how the sentence changes. Note that the model does not see pair “ENOpensubs”–“ENEuroparl” during training, so any difference is produced by the transfer learning effect – zero-shot translation. The aim of this experiment is to see if NMT can be potentially used to style adapt intra-language without additional mechanisms and whether there is ground for future research.

5 Experiments with Known Domains

In the following we describe the experiments using domain info as a tag and domain info as an embedding and compare the results to baseline and domain-tuning approach.

In the experiments we use mixed-domain parallel data consisting of Europarl (Koehn, 2005), OpenSubtitles2016 (Lison and Tiedemann, 2016), parallel data extracted from English-Estonian Wikipedia articles and some more mixed parallel corpora from the OPUS collection (Lison and Tiedemann, 2016). The size of the corpora is shown in Table 1. For each corpus we use a randomly chosen and held-out test set of $\tilde{3}000$ parallel sentences.

Table 1. Data sizes for the training data.

Corpus	Sents	EN tok	ET tok
Opensubs	10.32	83.57	67.56
Europarl	0.644	17.18	12.82
Wiki	0.135	2.281	2.089
Other	7.972	169.9	143.5
Total	19.07	272.9	225.9

Number of tokens (tok) given in Table 1 above is pre-BPE. All of the numbers are given in millions

5.1 Mining Wikipedia for Translations

Wikipedia[2] itself is a big set of articles. The articles have two properties, which are extremely useful from this thesis' task point of view. Firstly, the articles have links to the articles of same topic, but in different languages, which makes it easier to find comparable data from which to extract parallel data. Secondly, each article has one or several categories attached to it. This means that hypothetically domains can be assigned to at least some of the articles based on these categories.

Since Wikipedia is not traditionally used in MT as a parallel dataset, at least not for EN-ET language pair, data extraction and processing has to be done prior to the usage of the data.

To extract meaningful text from the Wikipedia XML dumps, we use the WikiExtractor tool[3]. We extract the data in a way that preserves article and paragraphs boundaries. The extraction is done separately for English and Estonian version.

[2]http://www.wikipedia.org/

[3]https://github.com/attardi/wikiextractor

After extracting text from the dumps, we apply another custom-made solution to detect parallel articles. The number of Wikipedia articles in English is well over 5 million whereas for Estonian it is just over 100 thousand. All Estonian articles are kept and only those English articles that have a parallel article in Estonian articles. This leaves us with roughly 70 thousand English articles.

The parallel articles form a comparable corpus. In case of this comparable corpora it is know that the articles are parallel in terms of topics but not in sentences. To extract parallel sentences from parallel articles, the LEXACC (Stefănescu et al., 2012) tool is used, which is a part of the ACCURAT toolkit (Pinnis et al., 2012; Skadiņa et al., 2012). Parallel sentence identification allows also to maintain the info of article origin, which means that direct domain assigning is possible. The identification process also assigns score to each sequence pair, which allows the creation of parallel sets with different grade of purity. The optimal grade of purity produced 340 thousand parallel sentences. The size of Estonian Wikipedia in total is 2.8 million sentences. To the rest 2.5 million sentences back-translation is applied to extend the Wikipedia dataset for EN-ET direction; the back-translated sentences are also filtered based on attention weights (Rikters and Fishel, 2017) with a 50% threshold.

5.2 Technical Settings

In preprocessing we apply BPE segmentation (Sennrich et al., 2016a) in a joint learning scenario, learning from the input and the output, limiting the vocabulary to 65,000 entries. The acquired segmentation mostly corresponds to the linguistic intuition on frequent tokens (which are left intact) and medium-frequency tokens (which are split into compound parts or endings off stems); low-frequency tokens (also names, numeric tokens) are split into letters and letter pairs.

The NMT model we use is encoder-decoder with an attention mechanism (Bahdanau et al., 2014), implemented in Nematus (Sennrich et al., 2017). All settings (like embedding size, number of recurrent layers in encoder and decoder, etc.) are kept at their default values. Batch size in experiments is 50 sequences.

5.3 Results

For the **Baseline** experiment we first train a baseline modelon all the datasets together, and used for translation. Then in the **Tuned** approach we fine-tune the **Baseline** model to each corpus separately.

Baseline results using the BLEU score are given in Table 2, along with out-of-domain translation results to show how the models are only suitable to translate one domain. Displayed corpora are: Eu–Europarl, Op–Opensubs, Wi–Wikipedia. Columns indicate the corpus to which the model in translation was tuned. Rows indicate the corpus that is used in translation. Best scores for each corpus are shown in bold. Scores lower than baseline are shown in gray. The expected behaviour is that for each corpus the best score is for the model that was tuned to the corpus i.e. **bold**. Second best should be the baseline, and all the rest should be lower than baseline i.e gray. The expected behaviour holds in this experiment also. The models were tuned to an optimal point where score of in-domain translation showed low improvement and other corpora started to deteriorate faster.

Table 2. BLEU scores for the English-Estonian direction all tuned scores.

Corp	**Base**	**Eu**	**Op**	**Wi**
Eu	24.37	**27.21**	13.82	12.01
Op	25.07	18.93	**25.34**	9.09
Wi	11.79	10.20	8.08	**12.83**

Table 3 and Table 4 show the BLEU scores and the p-values of the statistical significance of their difference for Baseline, fine-tuned baseline, domain tags, and domain feature approaches. For both domain tagging (**Tag**) and domain embedding (**Feat**), the **Baseline** translation approach is taken with the aforementioned modifications to the data; no fine-tuning step is performed.

For the comparability of the results, the number of iterations during training (800,000) and input parameters are kept equal for **Baseline**, **Tag**, **Feat**. The tuning of **Baseline** is done for additional 60,000 iterations. One iteration means one batch seen during training.

Table 3. BLEU scores and p-values for Estonian-English direction.

Corp	**Baseline**	**Tuned**	**Tag**	**Feat**
Eu	33.0±0.3	35.4±0.3	36.2±0.3	37.3±0.3
Op	27.9±0.6	28.1±0.6	30.5±0.6	30.3±0.6
Wi	15.3±0.4	15.4±0.4	16.9±0.4	17.7±0.4

Corp	**Baseline**	**Tuned**	**Tag**	**Feat**
Eu	0.0001 / 0.0001	0.009 / 0.0001	- / 0.0001	0.0001 / -
Op	0.0001 / 0.0001	0.0001 / 0.0001	- / 0.1	0.1 / -
Wi	0.0001 / 0.0001	0.0001 / 0.0001	- / 0.001	0.001 / -

The **Baseline** model in the Table 3 is trained without domain tags. **Tuned** is achieved by tuning these models with the specific corpus. **Tag** is trained with data that has domain tag prepended to each source sentence. **Feat** is trained with data that has domain embedding added as a feature to each source sequence word. p-values are given for significance against **Tag** and **Feat** respectively, separated with /.

Table 4. BLEU scores and p-values for English-Estonian direction.

Corp	**Baseline**	**Tuned**	**Tag**	**Feat**
Eu	22.5±0.3	25.3±0.3	25.4±0.3	24.9±0.3
Op	24.2±0.6	24.5±0.6	24.8±0.6	25.3±0.6
Wi	11.8±0.4	12.1±0.4	12.5±0.3	12.8±0.4

Corp	**Baseline**	**Tuned**	**Tag**	**Feat**
Eu	0.0001 / 0.0001	0.3 / 0.04	- / 0.04	0.04 / -
Op	0.01 / 0.001	0.09 / 0.03	- / 0.06	0.06 / -
Wi	0.01 / 0.001	0.06 / 0.03	- / 0.14	0.14 / -

Baseline model in Table 4 is trained without domain tags. **Tuned** is achieved by tuning these models with the specific corpus. **Tag** is trained with data that has domain tag prepended to each source sentence. **Feat** is trained with data that has domain embedding added as a feature to each source sequence word. p-values are given for significance against **Tag** and **Feat** respectively, separated with /.

Results show that both of the additional domain info models perform really well. The domain tag (**Tag**) model outperforms both of its baseline (**Baseline**) and tuned (**Tuned**) counterpart in ET–EN direction. It even goes as far as exceeding the **Tuned** approach by more than 1.0 BLEU in all domains. The same holds, but even more strongly, for the version where the domain embedding is added as an input feature for each word (**Feat**).

For EN–ET direction the results do not show such strong improvements. In this direction both **Tag** and **Feat** outperform **Baseline** for all domains. However, the scoring is quite close to the **Tuned** approach with the results between **Tag** and **Feat** also being closer than in ET–EN case. All in all, the fact that the domain tagging results are essentially on-par with Tuned approach, means it is superior to the **Tuned** approach in practice because of the fact that it requires only one model rather than three.

Table 5 shows an example of improved **ET–EN** translation. This is an example of improvement by removing the characteristic translation repetition ("only one can only be one") in **Tag** and **Ref** compared to **Base**. Also the **Feat** translation feels like a better mapping of **Ref** than translation of **Tag**. Additionally we observed improvements in both **Tag** and **Feat** like not missing out on adverbs and using rare words in translation, for example "pertinent" instead of "relevant". Since the quality of **Tuned** is close to **Tag** and **Feat**, it is omitted from the comparison since the differences would be highly circumstantial and would not hold much information in small scale.

Table 5. An example of Europarl corpus translations from Estonian to English using the Baseline, **Tag** and **Feat** methods.

Src (ET)	*vastuseid saab muidugi olla ainult üks : lõpetada kohe igasugused läbirääkimised Türgiga .*
Ref (EN)	*there is , of course , only one possible response : to immediately cease all negotiations with Turkey .*
Base (EN)	*only one can only be one : stop any negotiations with Turkey immediately .*
Tag (EN)	*the answer , of course , can only be one : stop all the negotiations with Turkey immediately .*
Feat (EN)	*there is , of course , only one answer : to put an end to all negotiations with Turkey immediately .*

6 Experiments with Automatic Domains

The methods applied in Section 5 work well, but these require pre-existing domain knowledge. For many scenarios however, previous domain knowledge does not exist, rendering the methods unusable. In the following section to overcome this problem, we propose to produce an automatic domain assignment. To test the approach, we conduct experiments with two data settings.

In the first experiment, we consider a single heterogeneous text domain. We apply both supervised and unsupervised tagging on this single text domain based on sentence vectors.

In the second experiment, we use texts from several domains but ignore the pre-specified text domains and replace these with automatic clustering based on sentence embeddings.

For both of the scenarios we take the domain tagging approach: even though domain features show better results, domain tags are more generic and compatible with any NMT architecture.

6.1 Automatic single-domain tagging

Text domains are often very diverse, consisting of many sub-domains. In the following we experiment if automatically detecting these sub-domains can improve translation. In this experiment Wikipedia is chosen as the non-homogeneous text domain. To make use of multi-domain approach, sub-domains need to be mined from the data and sentences annotated with the resulting clusters. To receive the annotation, two different approaches are taken.

Supervised approach involves assigning categories to roughly 10,000 articles, for which it could be done with high certainty. Assigning categories to more articles is problematic, because the categories assigned in Wikipedia can often be misleading in terms of content. With this approach 5-domain distribution is most logical, decided based on the Wikipedia category hierarchy. Next we sentence-tokenized the articles and tag each sentence in the article with the domain tag based on the categories assigned to the article.

After tagging the sentences we train a FastText classification model with default settings and apply it to classify the rest of the sentences that were not classified based on the article categories.

Unupervised approach involves firstly sentence splitting the whole Estonian Wikipedia data that was left after the filtering of backtranslation. Secondly, we train a sent2vec model and use it to calculate sentence vectors in an unsupervised manner. After that we apply unsupervised KMeans clustering to identify the clusters in the set of calculated sentence vectors. Finally, we tag each sentence with the label that it was assigned by KMeans. To find the optimal number of clusters, we create a version of 4, 5, 6, 8, and 12 clusters. The check for best number of clusters we do a sweep by training a model for each number of clusters. The input data for this is the whole Wikipedia corpus. The models are trained for 12 hours, which should be sufficient to make them diverge enough to choose the best number of clusters. We also train a regular model without data clustering for reference.

It is important to note that for this experiment a different test set is used than in the full data experiments. Thus the scores in Table 6 are not comparable to scores presented earlier.

Table 6. Dev set BLEU scores for Unsupervised Wikipedia parameter setting.

NClust	**C4**	**C5**	**C6**	**C8**	**C12**	**Ref**
BLEU	19.7	19.5	19.6	19.5	20.0	17.9

The initial sweep indicates that the best option for the unsupervised classification is 12 clusters. Also, the 12 hours – 100,000 iterations are already showing the effect that domain tagging has over the regular reference approach, making other clusterings also a viable choice.

Wikipedia Translation Results

In the final experiment, three models are trained:

- Supervised 5-domain source tag model
- Unsupervised 5-domain source tag model
- Unsupervised 12-domain source tag model
- Regular not domain-tagged model

Unsupervised 5-domain model is included to compare the performance of supervised and unsupervised approach with the same amount of domains, giving an indication of the "goodness" of these cluster assignments. The Unsupervised 12-domain model is included to compare the performance of best unsupervised clustering and the intuitively optimal supervised clustering. Supervised 12-domain model is not presented because it was not possible to produce such reasonable structure from ET Wikipedia. The results are presented in Table 7. The models were trained for 48 hours.

Table 7. BLEU scores and p-values for test on Wikipedia-only data to compare the effect of Unsupervised clustering (**Usup12, Usup5**), supervised clustering (**Super**) and no-clustering baseline approach (**Base**).

NClust	**Base**	**Usup12**	**Usup5**	**Super**
BLEU	23.6±0.4	26.0±0.4	25.2±0.4	25.8±0.4
pBase	-	0.0001	0.0001	0.0001
pU12	0.0001	-	0.01	0.1
pU5	0.0001	0.01	-	0.03
pSup	0.0001	0.1	0.03	-

The p-values in Table 7 are shown in respect to the version where the value is **-**.

As shown in Table 7, the Supervised approach (**Super**) with five clusters slightly outperforms Unsupervised 5-cluster approach (**Usup5**). The best option for Unsupervised clustering (**Usup12**) performs as well as the Supervised approach. The results show that Unsupervised approach is comparable in performance to the Supervised approach, which means that at least in this setting both of the approaches are viable. Even more so, when obtaining labelled data for supervised clustering can often require a lot of additional effort, the unsupervised approach is not chained by the (lack) of pre-existing knowledge about the data.

Most important is the fact that both of the unsupervised cluster versions outperform the regular reference (**Ref**) version where sentence cluster tags were not used. This shows that the unsupervised clustering approach can potentially be used in settings that previously were viewed upon as single clusters. For example OpenSubtitles corpus could be clustered further, to improve the translations.

6.2 Unsupervised multi-domain tagging

Hinging on the fact that domain tagging approach outperformed the traditional tuning approach and on the results that unsupervised Wikipedia dataset clustering produced, the "traditional" approach of text domains should be given another look. One possible action is to cluster or sub-cluster the existing parallel data to restructure it from the domain point of view.

In addition to the results produced on wikipedia dataset, the hypothesis on why this would work, is that large text domains are probably not very homogeneous. Also, different domains have probably pretty big overlap of similar sentences. This would mean that the usual approach of domain tuning or domain tagging does not achieve its true potential, because predefined domains are *de facto* several domains and the same domains are actually present in other predefined domains also.

To check for this property and its potential benefit for NMT, we cluster existing parallel sentences of known domains together to n clusters in the previously described unsupervised manner. On this data, we train NMT models with domain tagged sentences, and finally, cluster the test set sentences in a supervised manner with a supervised clustering model that is trained on the data obtained from unsupervised clustering.

The training is done using Nematus with the same settings as in the initial experiment with domain tags. Firstly, we do a sweep of clusters by training 4, 8, 16, and 32 cluster versions for both EN–ET and ET–EN direction. After that we choose the version that has achieved the best BLEU scores on the dev sets for both of the directions and train it for the same number of iterations and with same parameters as in the initial domain tag experiment with full data.

Results of unsupervised multi-domain tagging

To evaluate the model performance, we train a supervised FastText classification models on the tagged training data. We apply these models on the test/dev sets to classify the sentences. This means that each of the sets – Opensubs, Europarl, and Wiki – gets actually tags from several clusters, depending on which cluster the FastText model assigns to each of the sentences. This means that for each source test set we create four different versions, each for cluster numbers 4, 8, 16, and 32.

The initial parameter sweep shows that the best option is 16 clusters for both EN–ET direction in Table 8 and ET–EN direction in Table 9 across all test sets. Hence the final models are both trained with 16 clusters.

Table 8. BLEU scores for English-Estonian direction sweep.

Corp	**C4**	**C8**	**C16**	**C32**
Eu	4.13	3.19	**5.94**	4.17
Op	9.41	9.36	**10.80**	10.62
Wi	1.09	0.94	**1.31**	0.81

The model used in Table 8 is trained on parallel data that is tagged in unsupervised manner using sent2vec + Kmeans clustering. The dev sets are clustered based on this tagged data using FastText. The best scores for each corpus are presented in **bold**.

Table 9. Test set BLEU scores for Estonian-English direction sweep.

Corp	**C4**	**C8**	**C16**	**C32**
Eu	20.48	19.88	**20.82**	18.43
Op	20.05	19.54	**20.17**	20.01
Wi	4.61	4.38	**5.50**	4.32

The model used in Table 9 is trained on parallel data that is tagged in unsupervised manner using sent2vec + Kmeans clustering. The dev sets are clustered based on this tagged data using FastText. The best scores for each corpus are presented in **bold**.

The final results, where the 16 cluster models are trained for the same amount of iterations as in the initial full data experiments, are presented in Table 10 and Table 11 for EN–ET and ET–EN language pairs respectively.

Table 10. Test set BLEU scores and p-values for English-Estonian direction.

Corp	**Baseline**	**Tuned**	**Tag**	**Unsup**
Eu	22.5±0.3	25.3±0.3	25.4±0.3	24.5±0.3
Op	24.2±0.6	24.5±0.6	24.8±0.6	24.6±0.6
Wi	11.8±0.4	12.1±0.4	12.5±0.3	11.1±0.4
Corp	**Baseline**	**Tuned**	**Tag**	**Unsup**
Eu	0.0001 / 0.0001	0.3 / 0.03	- / 0.004	0.004 / -
Op	0.01 / 0.03	0.09 / 0.4	- / 0.2	0.2 / -
Wi	0.01 / 0.01	0.06 / 0.005	- / 0.0001	0.0001 / -

Baseline model in Table 10 is trained without domain tags. **Tuned** is achieved by tuning these models with the specific corpus. **Tag** is trained with data that has domain tag prepended to each source sentence. **Unsup** is trained with data that has domain tags assigned to each sentence in an previously described unsupervised manner. p-values are given for significance against **Tag** and **Unsup** respectively, separated with **/**.

The results show that the unsupervised clustering approach performs similarly to the pre-defined tag version with only some performance issues. In case of EN–ET translation the unsupervised approach performs worse than **Tuned** and **Tag** for Europarl and worse than all other three for Wikipedia. In case of ET–EN the unsupervised approach performs slightly worse than **Tag** for Wikipedia, but outperforms other versions.

Since the clustering approach is pretty much applied out-of-the-box, then improved clustering could provide considerable improvements. This means that with improved clustering the unsupervised tagging approach can serve as a viable alternative to the pre-defined domain tagging approach.

Our hypothesis is that the good results of unsupervised approach are caused by the pre-defined domains being less homogeneous in content than expected. Also, since the unsupervised clustered "domains" produce similar results, we expect the domain homogeneity to be similar to that of pre-defined clusters. In the following section we explore the pre-defined domain and unsupervised cluster structures and the relations between them to validate these hypotheses.

Table 11. Test set BLEU scores and p-values for Estonian-English direction.

Corp	Baseline	Tuned	Tag	Unsup
Eu	33.0±0.3	35.4±0.3	36.2±0.3	36.0±0.3
Op	27.9±0.6	28.1±0.6	30.5±0.6	30.2±0.6
Wi	15.3±0.4	15.4±0.4	16.9±0.4	16.0±0.4
Corp	**Baseline**	**Tuned**	**Tag**	**Unsup**
Eu	0.0001 / 0.0001	0.009 / 0.01	- / 0.3	0.3 / -
Op	0.0001 / 0.0001	0.0001 / 0.0001	- / 0.1	0.1 / -
Wi	0.0001 / 0.004	0.0001 / 0.009	- / 0.01	0.01 / -

Baseline model in Table 11 is trained without domain tags. **Tuned** is achieved by tuning these models with the specific corpus. **Tag** is trained with data that has domain tag prepended to each source sentence. **Unsup** is trained with data that has domain tags assigned to each sentence in an previously described unsupervised manner. p-values are given for significance against **Tag** and **Unsup** respectively, separated with /.

Unsupervised cluster structure

The main question for the unsupervised approach is, how does unsupervised clustering differ from pre-defined clustering. To examine this, we produce t-SNE figures to check how placements in unsupervised projections differ from the predefined ones. Each dot on the figures stands for the two-component projection on one test sentence vector. The sentence vectors are calculated using unsupervised sent2vec models trained on the training data, since these are independent from both of the clusterings. Also, in the following figures each color stands for separate cluster. In case of the pre-defined clusters we just color the dot according to the origin corpus. For unsupervised approach, we tag the test sets using FastText models trained on tagged train set and color these accordingly. The unsupervised cluster differences with pre-defined clusters can be examined in the following Figure 2 and Figure 3. Figure 2 shows that the difference between unsupervised and predefined label coloring is quite extensive. Firstly, the clusters seem to be less disperse and with smaller overlap in predefined coloring case compared to unsupervised clustering. Also, because unsupervised approach has more clusters, the data is clustered into further parts in unsupervised case. Similar effects can be examined in Figure 3 for Estonian. These figures indicate that the current version of unsupervised sentence clustering does produce viable clusters,but the pre-defined clusters are still more homogeneous. This is probably the reason why pre-defined domain tagging in most cases slightly outperforms the unsupervised clustering. However, since the unsupervised clustering in this case is applied out-of-the-box, there is plenty of room for improvements. Also, since pre-defined clusters still have a considerable overlap in the projection, the unsupervised clustering should be able to surpass the pre-defined clustering in cluster separation and homogeneity.

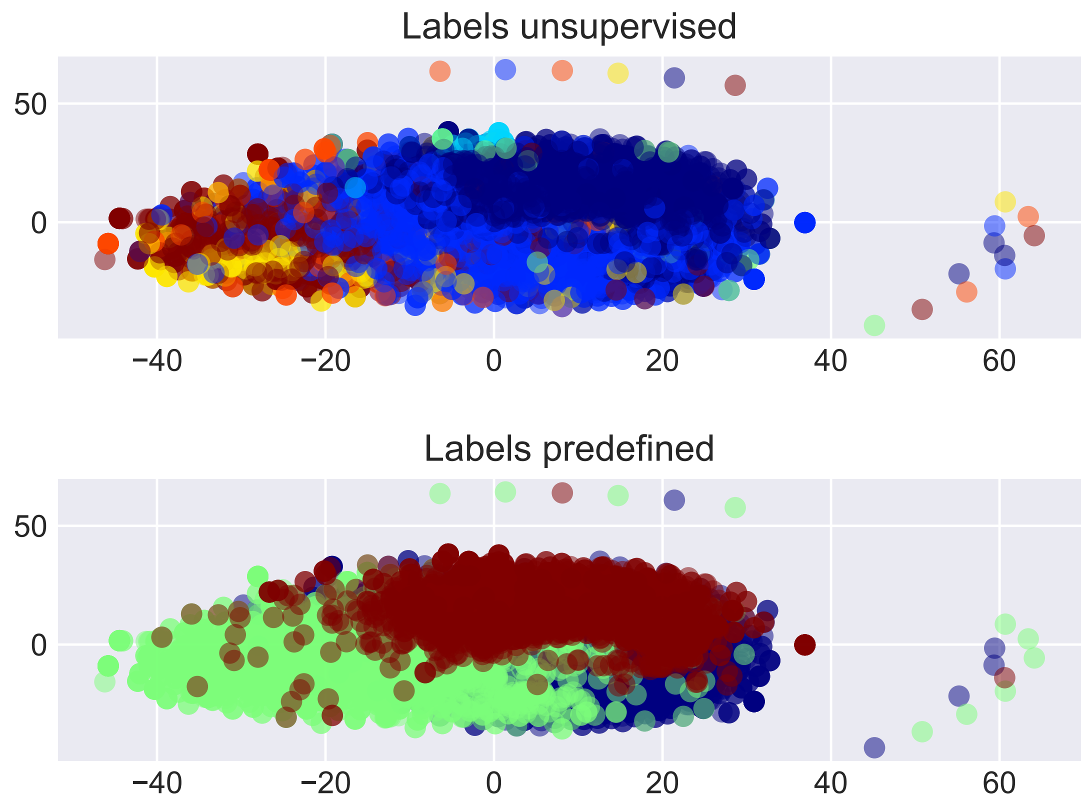

Figure 2. The t-SNE 2-component projections of English joint test set sent2vec sentence vectors, coloured based on unsupervised clusters and same joint test set sentence vectors coloured based on predefined domains.

To examine the cluster differences in more precise numbers, we display the unsupervised cluster sharing of different domains in Table 12 and Table 13. The cluster sharing shows that different corpora actually share similar sequences, at least from sentence vectors point of view. For example in Table 13 **C1** is prominently present in both Europarl (Eu) and Wikipedia (Wi), however almost non-existent in Opensubs. **C15** however is present in all three. Similar behaviour can be examined in Table 12. The sharing structures indicate that the hypothesis of pre-defined domains consisting of several sub-domains holds. Also, it is visible that many of these domains are present in most or all of the corpora used, meaning that unsupervised domain tagging could actually produce tags that are more descriptive of the sequence than the tags originating from pre-defined domains.

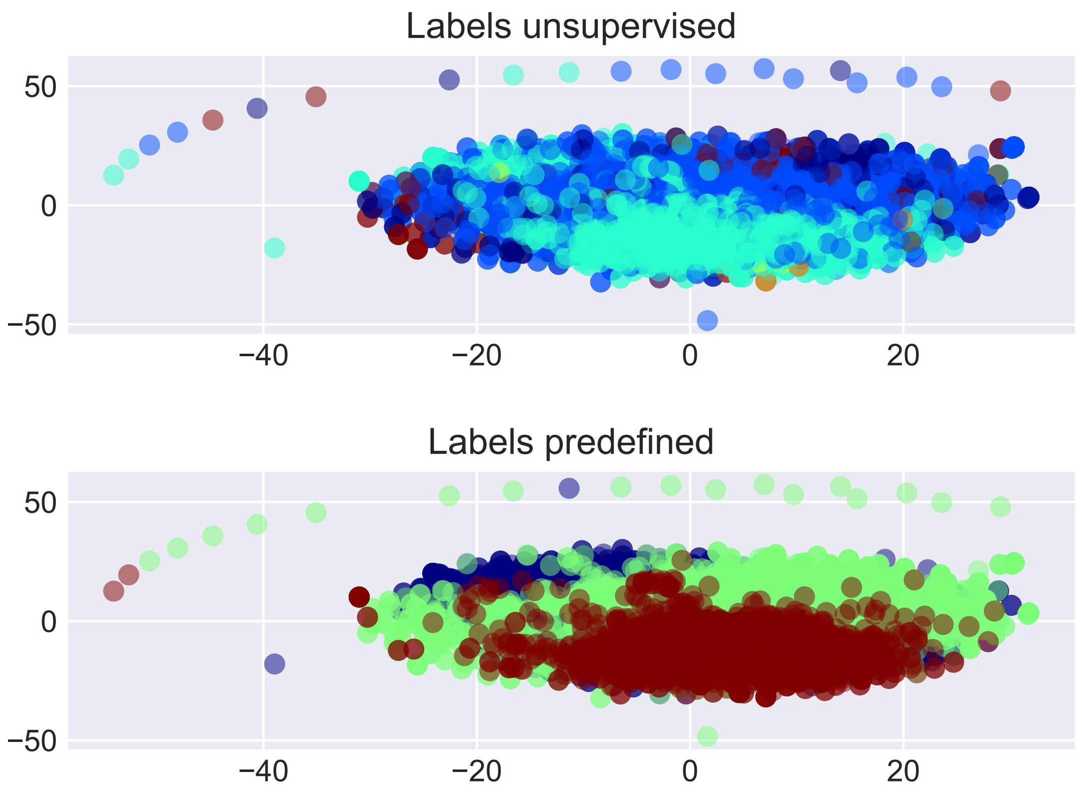

Figure 3. The t-SNE 2-component projections of Estonian joint test set sent2vec sentence vectors, coloured based on unsupervised clusters and same joint test set sentence vectors coloured based on predefined domains.

Additionally, cluster structure differences between languages can be examined. If the clusters for both English and Estonian were aligned, the columns in both tables would also align more or less with one another. However, there are visible differences. Firstly, parallelity for Europarl and Wikipedia is swtiched in Estonian compared to English. The largest cluster of Europarl corresponds to second largest in Wikipedia and vice versa in Estonian, but in English these are aligned. Secondly, in Estonian, Opensubs has one main cluster, whereas is English it has two. Cluster structure differences between languages are further illustrated in the following t-SNE Figure 4. The plot shows that for both Estonian and English distinct clusters exist with some overlap. The cluster structure however is different between languages.

All this means that the domain structure from sentence vector point of view is different for English and Estonian, which means it is probably reasonable to apply clustering to these separately. However, the possibility of joint clustering should be investigated in

Table 12. Unsupervised cluster sharing between different domains in English.

Corp	**C14**	**C12**	**C9**	**C3**	**C6**	**C15**	**C1**
Eu	2417	430	42	22	21	16	2
Os	18	1331	-	181	1015	7	398
Wiki	1056	1811	4	-	9	69	1

Table 13. Unsupervised cluster sharing between different domains in Estonian.

Corp	**C1**	**C15**	**C12**	**C6**	**C8**	**C2**
Eu	2469	431	26	11	9	4
Os	21	2282	364	-	282	1
Wiki	1732	1212	2	2	2	-

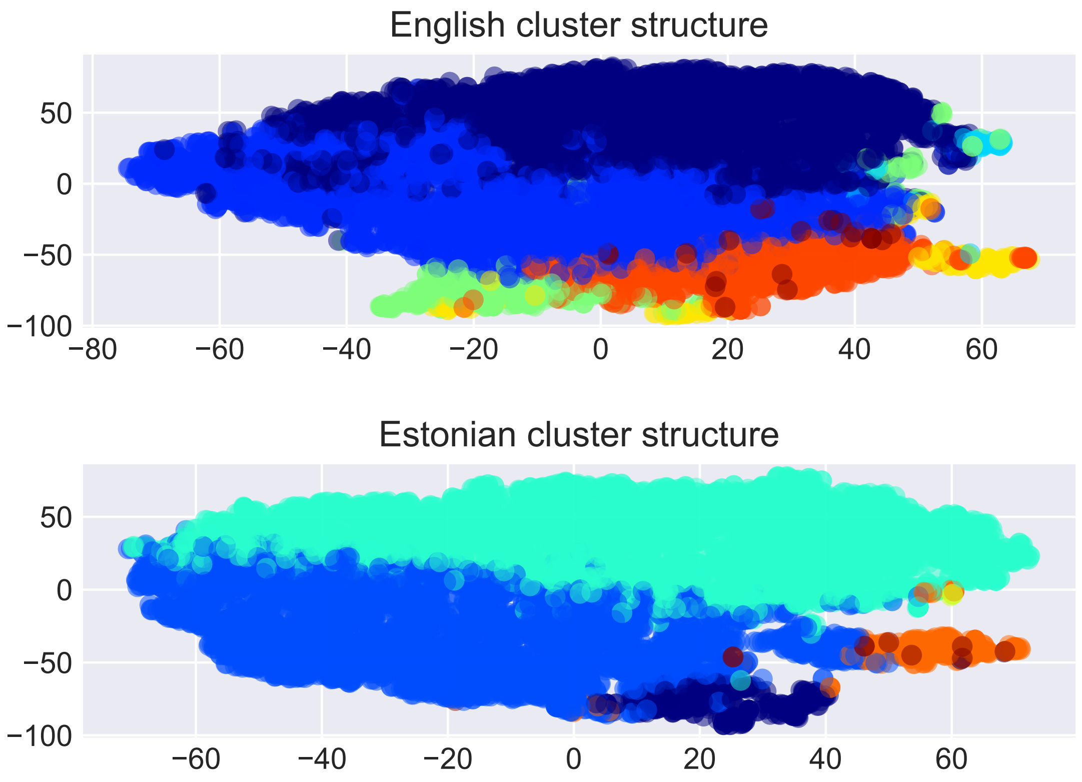

Figure 4. The t-SNE 2-component projections of Estonian joint test set sentence vectors and English joint test set sentence vectors, both coloured based on unsupervised clusters.

future work.

To further examine the unsupervised clustering, we produce some intra-domain assessments. Since we do not use backtranlsated data in the multi-domain tagging experiments to reduce the variable components, the Wikipedia dataset with 135,000 parallel sentences is quite small part of the full data. Thus the integral part of the following analysis is done on the OpenSubtitles corpus, which is the biggest single corpus in the data and probably also consists of several sub-domains.

Table 14 displays the OpenSubs test sets cluster structure. The test sets are tagged using FastText models trained on tagged train set. One can see that different train set clusters produce different granularity in test sets also. For **C4, C8** the OpenSubs structure is similar, same holds for other test sets. **C16** vs **C8** however shows a significant difference in test set clustering. Here we see that OpenSubs, which based on content is probably not homogeneous domain, is separated quite granularly in **C16**, producing 3–4 main sub-domains. In **C32** the test set is clustered even further, but based on sweep scores, it could be said that the achieved clustering is already too granular.

Table 14. Cluster structure of FastText tagged English OpenSubs test sets.

Corp	**N1**	**N2**	**N3**	**N4**	**N5**	**N6**
C4	2921	29	-	-	-	-
C8	2907	43	-	-	-	-
C16	1331	1015	398	181	18	7
C32	1137	828	356	293	241	71

The test sets in Table 14 are clustered based on tagged train data. The clusters are numbered left to right based on size. Here only top 6 clusters are shown. For C32 $N7 = 11$, $N8 = 6$, $N9 = 3$, $N10 = 2$, $N11 = 2$. Test set structures for Estonian sets are similar.

Considering that the used OpenSubs cluster is 10 million sentence pairs in size, it can be said that **C16** finds 5 significant sub-domains and one less significant sub-domain inside it. This shows that, at least from sentence vectorizing point of view, there exists more than one domain inside OpenSubs, and similarly in other domains.

The domain structure of English OpenSubs is further visualised in Figure 5. The plots are 2-component t-SNE projections of Opensubs test set sentence vectors. The sentence vectors are produced using previously trained FastText model. On the plots, the axes are the components of t-SNE projections. As we see with the option of 8 clusters and 4 clusters, the test set is clustered almost entirely as one domain. The smaller sub-domain (**N2** in Table 14) is visible on both plots as occasional blue dots. For 16 and 32 clusters we see already distinguishable clusters, to show that OpenSubs texts consist actually of many sub-domains.

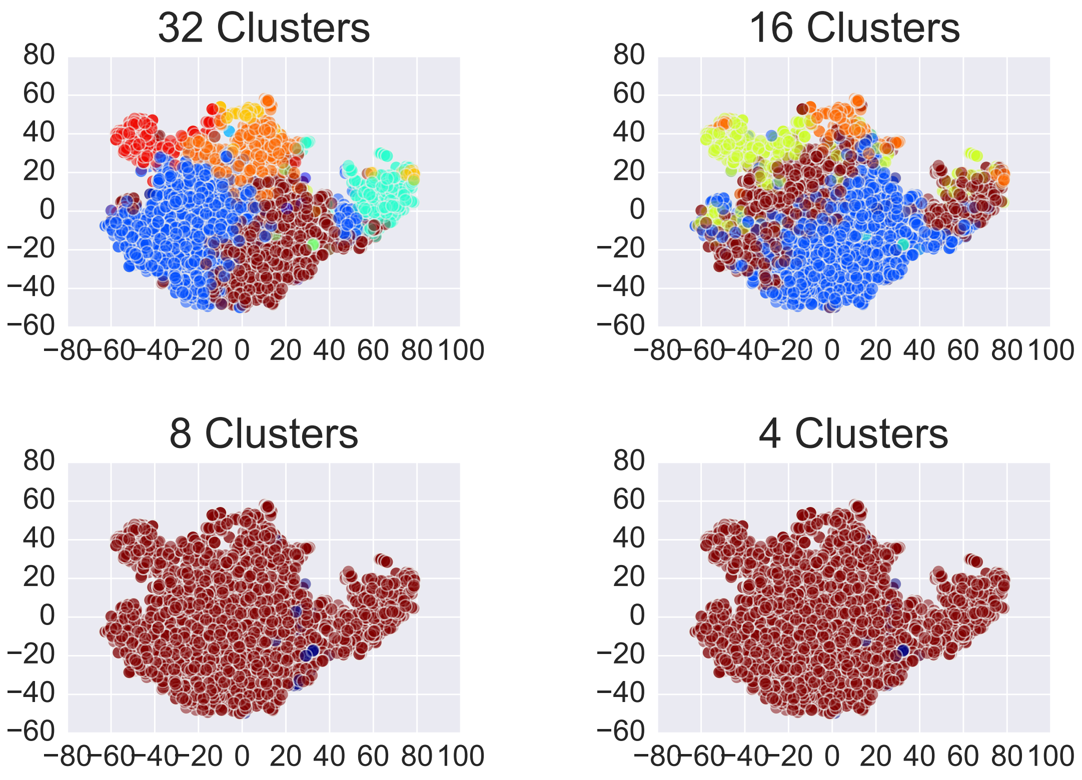

Figure 5. The t-SNE 2-component projections of English Opensubs sentence vectors, coloured based on unsupervised clusters.

Interestingly for 16 cluster option two of the big clusters - red and blue - overlap in large part. For 32 cluster option the cluster are however more separate and on visual inspection seem better. However, this does not come out in the results, which raises the question of why 16 clusters produces better results than 32 clusters. The question should be investigated in future work. One possible course of action is to train 32 cluster option also fully and see, how much the results differ between 16 and 32 clusters on fully trained models.

Table 15 displays the English Europarl test sets cluster structure. The test sets are tagged using FastText models trained on tagged train set. Unlike the Opensubs cluster structure in Table 14, the Europarl cluster structure is much more compact. For each number of clusters a distinct main cluster and a second slightly smaller one exists, which indicates that Europarl as a corpus is more homogeneous that Opensubs. The difference between larger number of clusters is that the main cluster gets some small parts "chipped off". To examine the effect further the t-SNE 2-component projections of English Europarl sentence vectors are presented on Figure 6. The figure shows that

Table 15. Cluster structure of FastText tagged English Europarl test sets.

Corp	**N1**	**N2**	**N3**	**N4**	**N5**	**N6**
C4	2463	486	1	-	-	-
C8	2614	316	20	-	-	-
C16	2469	431	26	11	9	4
C32	2270	482	60	54	30	21

The clusters in Table 15 are numbered left to right based on size. Here only top 6 clusters are shown. For C32 $N7 = 15$, $N8 = 10$, $N9 = 3$, $N10 = 2$, $N11 = 2$, $N12 = 1$. Test set structures for Estonian sets are similar.

the cluster projections of Europarl are much more disperse than those of Opensubs. This indicates two things. Firstly, Europarl really seems to be more homogeneous that Opensubs. Secondly, the fact that Europarl is 0.6 million parallel sentences in size, whereas OpenSubs is 10 million parallel sentences, could mean that the produced sentence vectors are tuned more to separate OpenSubs sequences than Europarl sequences. This is another question in unsupervised tagging which should be investigated further in future work. One possible course of action would be to compare this version of clustering and a stratified version of clustering to see if the stratified clustering can produce better separation for all of the involved corpora than the current one.

TSNE 2-component projections for rest of the English and Estonian test sets are also calculated and these are presented in **Appendix C**. The cluster structure tables for corpora not displayed in this section, are presented in **Appendix A**

When looking at the number of clusters present in Table 14, one could notice that the clusters present is less than number of clusters defined. It should be kept in mind that 3 main text sources are used in training set and also fourth mixed-corpus which could be divided into 5-6 parts, so 8-9 text domains in total. Also, some sentences are quite distinct from the others based on full train set cluster structure as visible from the train set structure of **C16** in Figure 7. Train set structure of Estonian **C16** is presented in **Appendix B**

All-in-all, taking into consideration the fact that unsupervised approach allows new sentences to be translated with potentially more appropriate domain assigned to them, the unsupervised tagging approach can be seriously considered as the go-to approach for multi-domain translation models and with some improvement, it could actually surpass the pre-defined domain tagging in translation quality. The models trained using unsupervised tagging and tagging test sentences using FastText classification model are also integrated into open-source Neurotõlge translation service (Tars et al., 2017)[4].

[4]Specifically in Nazgul part of Neurotõlge. Documentation for Nazguls is on `https://github.com/`

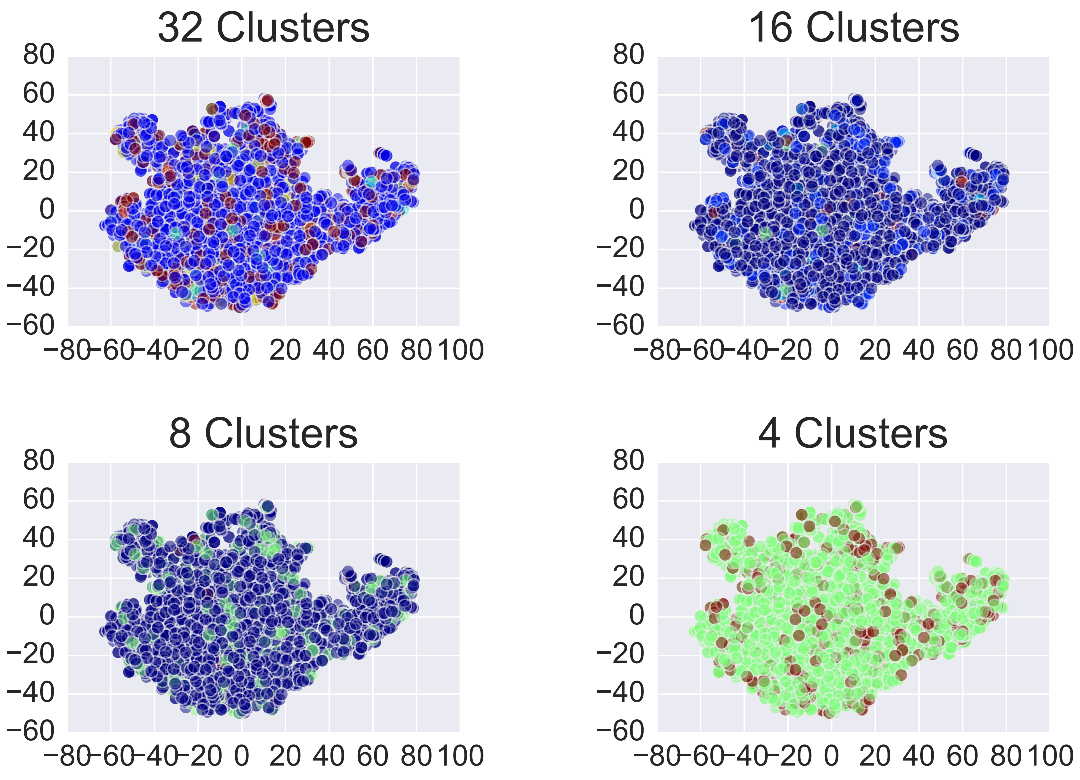

Figure 6. The t-SNE 2-component projections of English Europarl sentence vectors, coloured based on unsupervised clusters.

TartuNLP/nazgul

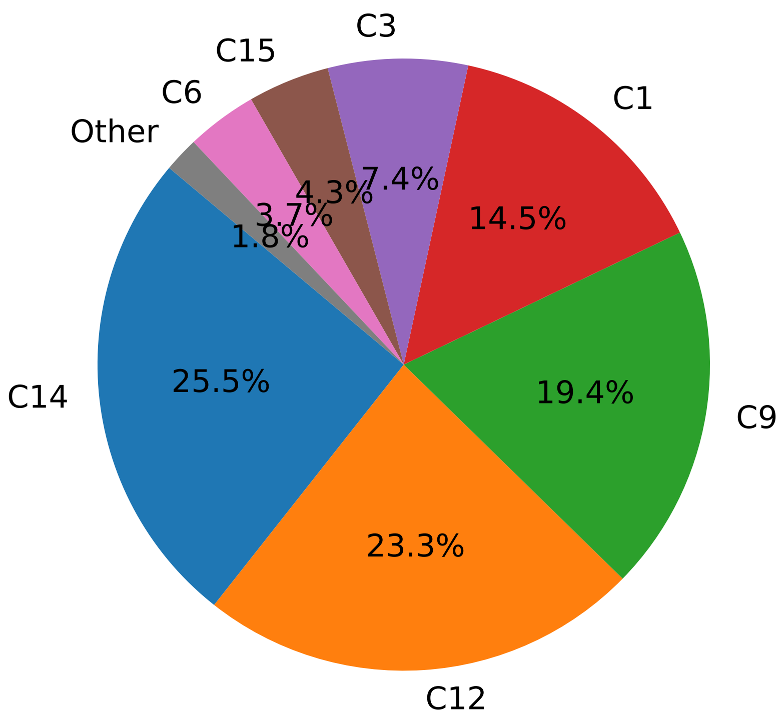

Figure 7. A piechart of English training set cluster proportions. The ’Other’ cluster contains nine smallest clusters.

7 Experiments with zero-shot style adaptation

In this following section we show the results of zero-shot style adaption. The aim of the approach is to explore the possibility of intra-language style adaptation to transform text into more appropriate form (e.g EN_Manchester to EN_formal) through zero-shot translation.

To test the possibility of zero-shot style adaptation, we train a EN–ET+ET–EN neural system. The system has both language and domain information included in the preceding tag.

In the test setting we take sentences from English test set and tag each with 2eneu (to english europarl), 2enos (to english opensubs), and 2enwi (to english wiki). The expected behaviour of the model would be that "translating" into the true tag, the sentence does not change while "translating" into the false tags, the sentence is more prone to change intra-language in style. Mostly the sentences remained the same, which is indicated by the BLEU scores of 86.7, 85.5 and 85.6 respectively, compared to the source text.

To observe the changes more thoroughly, we sample 100 sentences from the translation set. In general four types of changes are observed:

- Omission of a word or sequence of words
- Translation of a word into Estonian rather into another domain of English
- Transition from a short form to longer form or vice versa, e.g. **didn 't – did not**
- Single word replacements

In the following Table 16, Table 17, Table 18 some examples of these occurances are shown.

Table 16. Style transition from a short form to longer form.

Domain	**Sentence**
OS	*I didn 't !*
to_OS	*I didn 't !*
to_EU	*I did not !*
to_WI	*I did not !*

The example in Table 16 is a nice example of the style adaptation as transition from a short form to longer form of **"didn 't" – "did not"**. In subtitles domain the short forms are okay, whereas in formal corpora such as Europarl and Wikipedia, short forms are not acceptable. Of course this substitution has errors too, such as **"can 't" – "could not"**.

Table 17. Deletion of word sequence during style adaption translation.

Domain	**Sentence**
OS	*Dwight : how about you , cuz ?*
to_OS	*Dwight :*
to_EU	*Dwight : how about you , cuz ?*
to_WI	*Dwight :*

Table 18. Translation of a word into Estonian rather into another domain of English.

Domain	**Sentence**
WI	*the term sciatica describes a symptom ...*
to_OS	*the term sciatica describes a sümptom ...*
to_EU	*the term sciatica kirjeldab a sümptom ...*
to_WI	*the term sciatica describes a sümptom ...*

In example Table 17 two of the domains delete the sequence that follows ":". Interestingly enough this occurs also in the original domain. Probably due to the training data profile.

In example Table 18 all of the domain transitions translate **"symptom"** into **"sümptom"**, which is an accurate translation into Estonian, but not the intended outcome. Europarl also translates (accurately) **describes** to **kirjeldab**.

All in all it can be said, that yes, to some extent the intra-language domain adaptation works, but with our experimental setting it produces more noise than usable output. This effect could be alleviated by training the model more and having data that is more representative of the involved domains. However, it is probable that those modifications would need some additional mechanism to produce usable results.

8 Conclusions

The results from the experiments - EN–ET and ET–EN direction parallel translation, Wikipedia data translation, and unsupervised sentence tagging - show that both of the two chosen multi-domain approaches outperform regular approach of uniform translation and domain-tuning.

This indicates the hypothesis that the parameter sharing effect discussed in Google's zero-shot article would benefit domain translation holds. The translation scores even outperform domain fine-tuning approach, which could be explained by the same parameter sharing. In fine-tuning the model is tuned to translate sentences characteristic to the domain it has been tuned to. This means that domain-characteristic sentences get translated really well. On the other hand, the not-so-characteristic sentences get neglected. The parameter sharing effect of the multi-domain approach helps negate the negative effect by the support of other domains while still learning to more effectively represent each domain by the additional domain info.

Furthermore, the results indicate that adding domains as an input feature can have even stronger effect on the translation scores. This shows that concatenating the domain feature embedding with word embedding at each timestep - basically remembering the source domain equally throughout the sequence improves model performance. This could be explained by the fact that in tag prepending case, the neural net may "forget" for longer sequences what the input tag was, making the effect of it weaker. However, the differences in scores are not drastically different from the domain tag prepending. This means that for the sake of data simplicity and model simplicity the tag prepending approach could prove more reasonable of the two for in-production settings. Also, unlike domain embeddings, the tag prepending approach is compatible with many modern NMT architectures, for example Transformers.

Additionally, to expand the domain tagging approach to texts without domain knowledge, we produced an approach of automatically detecting clusters in text corpora. The performance of unsupervised domain tagged model indicates that the approach could possibly substitute the pre-defined domain tagging approach. The unsupervised tagging certainly serves as an improvement in less homogeneous single domain settings, where we showed on the example of Wikipedia that the sub-domain detection provides a clear improvement compared to baseline no-tagging version in translation scores.

No less important are the facts that the unsupervised tagging approach can ensure better domain assignment to each new sentence and can efficiently incorporate new data from various small domains to fortify each of the learned "domain" (clusters). We noted considerable differences in cluster distribution between pre-defined defined domains and automatically detected domains, with the latter being less separated in the cluster structure, but promising approach with improvements to the clustering approach. It has to be taken into account that the unsupervised clustering performed in these experiments is applied basically in out-of-the-box manner, which means that domain assignments can be improved and thus the translation scores should also improve.

Finally, the use of the domain tagging approach is explored in multilingual setting and the possibility of intra-language domain adaptation through zero-shot translation. The results show clear evidence of this kind of model being able to domain-adapt text, however, with small but visible text deterioration.

For future work the clustering in fully unsupervised tagging approach should be improved to see if this gives a visible improvement in translation scores.

Secondly, a more comprehensive sweep on number of clusters should be done. It would be interesting to see for how many clusters the effect still persists. This however would need more extensive computational resources and should probably be done with some model dataset.

The differences of the two approaches - source sentence tagging and adding domain info as an input feature - deserve to be looked into more deeply. More precisely, the result profiles of the two in different cases of domain granularity.

Finally, in this work domains are still treated as nominal values; it would be interesting to explore the estimation of domain embeddings at translation time as continuous values.

Acknowledgements

This work was supported by the Estonian Research Council grant no. 1226 and Study-ITin.ee.

Bibliography

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014. URL `http://arxiv.org/abs/1409.0473`.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *CoRR*, abs/1607.04606, 2016. URL `http://arxiv.org/abs/1607.04606`.

Denny Britz, Quoc Le, and Reid Pryzant. Effective domain mixing for neural machine translation. In *Proceedings of WMT*, pages 118–126, Copenhagen, Denmark, 2017.

Wenhu Chen, Evgeny Matusov, Shahram Khadivi, and Jan-Thorsten Peter. Guided alignment training for topic-aware neural machine translation. *CoRR*, abs/1607.01628, 2016. URL `http://arxiv.org/abs/1607.01628`.

Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D14-1179`.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9 (8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `http://dx.doi.org/10.1162/neco.1997.9.8.1735`.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's multilingual neural machine translation system: Enabling zero-shot translation. *CoRR*, abs/1611.04558, 2016. URL `http://arxiv.org/abs/1611.04558`.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759, 2016. URL `http://arxiv.org/abs/1607.01759`.

Catherine Kobus, Josep Crego, and Jean Senellart. Domain control for neural machine translation. In *Proceedings of RANLP*, pages 372–378, Varna, Bulgaria, 2017.

Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of MT Summit*, volume 5, pages 79–86, Phuket , Thailand, 2005.

Pierre Lison and Jörg Tiedemann. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of LREC*, pages 923–929, Portorož, Slovenia, 2016.

Minh-Thang Luong and Christopher D Manning. Stanford neural machine translation systems for spoken language domains. In *Proceedings of IWSLT*, pages 76–79, Da Nang, Vietnam, 2015.

Robert Östling and Jörg Tiedemann. Continuous multilinguality with language vectors. In *Proceedings of EACL*, pages 644–649, Valencia, Spain, 2017.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. Unsupervised learning of sentence embeddings using compositional n-gram features. *CoRR*, abs/1703.02507, 2017. URL `http://arxiv.org/abs/1703.02507`.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318, Philadelphia, Pennsylvania, USA, 2002.

Mārcis Pinnis, Radu Ion, Dan Ştefănescu, Fangzhong Su, Inguna Skadiņa, Andrejs Vasiļjevs, and Bogdan Babych. Accurat toolkit for multi-level alignment and information extraction from comparable corpora. In *Proceedings of ACL*, pages 91–96, Jeju Island, Korea, 2012.

Matīss Rikters and Mark Fishel. Confidence through attention. In *Proceedings of MT Summit*, pages 299–311, Nagoya, Japan, 2017.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of ACL*, pages 1715–1725, Berlin, Germany, 2016a.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Controlling politeness in neural machine translation via side constraints. In *Proceedings of NAACL*, pages 35–40, San Diego, California, 2016b.

Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nadejde. Nematus: a toolkit for neural machine translation. In *Proceedings of EACL*, pages 65–68, Valencia, Spain, 2017.

Yusuke Shibata, Takuya Kida, Shuichi Fukamachi, Masayuki Takeda, Ayumi Shinohara, and Takeshi Shinohara. Byte pair encoding: A text compression scheme that accelerates pattern matching. 09 1999.

Inguna Skadiņa, Ahmet Aker, Nikos Mastropavlos, Fangzhong Su, Dan TufiÈ™, Mateja Verlic, Andrejs Vasiļjevs, Bogdan Babych, Paul Clough, Robert Gaizauskas, Nikos Glaros, Monica Lestari Paramita, and Mārcis Pinnis. Collecting and using comparable corpora for statistical machine translation. In *Proceedings of LREC*, pages 438–445, Istanbul, Turkey, 2012.

Dan Stefănescu, Radu Ion, and Sabine Hunsicker. Hybrid parallel sentence mining from comparable corpora. In *Proceedings of EACL*, pages 137–144, Trento, Italy, 2012.

Sander Tars, Kaspar Papli, Dmytro Chasovskyi, and Mark Fishel. Open-Source Neural Machine Translation API Server. *The Prague Bulletin of Mathematical Linguistics*, 109:5–14, October 2017. ISSN 0032-6585. doi: 10.1515/pralin-2017-0034. URL `https://ufal.mff.cuni.cz/pbml/109/art-tars-et-al.pdf`.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. URL `http://www.jmlr.org/papers/v9/vandermaaten08a.html`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf`.

Appendix A

This appendix displays the unsupervised cluster structures for test sets that were not displayed in the main part of the thesis. These test sets are Estonian Europarl, Estonian Wiki, Estonian Opensubs and English Wiki.

Table 19. Cluster structure of FastText tagged English Wiki test sets.

Corp	**N1**	**N2**	**N3**	**N4**	**N5**	**N6**
C4	2921	29	-	-	-	-
C8	2907	43	-	-	-	-
C16	1811	1056	69	9	4	1
C32	2184	523	110	80	21	20

The clusters are numbered left to right based on size. Here only top 6 clusters are shown. For C32 $N7 = 11$, $N8 = 1$.

Table 20. Cluster structure of FastText tagged Estonian Opensubs test sets.

Corp	**N1**	**N2**	**N3**	**N4**	**N5**	**N6**
C4	2503	465	12	-	-	-
C8	2486	478	15	1	-	-
C16	2282	364	282	21	1	-
C32	1271	1044	285	245	97	27

The clusters are numbered left to right based on size. Here only top 6 clusters are shown. For C32 $N7 = 7$, $N8 = 2$, $N9 = 1$, $N10 = 1$.

Table 21. Cluster structure of FastText tagged Estonian Europarl test sets.

Corp	**N1**	**N2**	**N3**	**N4**	**N5**	**N6**
C4	2175	747	28	-	-	-
C8	2302	614	28	6	-	-
C16	2469	431	26	11	9	4
C32	2173	628	55	42	20	17

The clusters are numbered left to right based on size. Here only top 6 clusters are shown. For C32 $N7 = 8$, $N8 = 5$, $N9 = 1$, $N10 = 1$.

Table 22. Cluster structure of FastText tagged Estonian Wiki test sets.

Corp	**N1**	**N2**	**N3**	**N4**	**N5**	**N6**
C4	1759	1219	2	-	-	-
C8	1503	1476	1	-	-	-
C16	1732	1212	2	2	2	-
C32	2087	777	97	15	2	2

Appendix B

This appendix displays the Estonian training set cluster proportions. The cluster assignment is produced in an unsupervised manner.

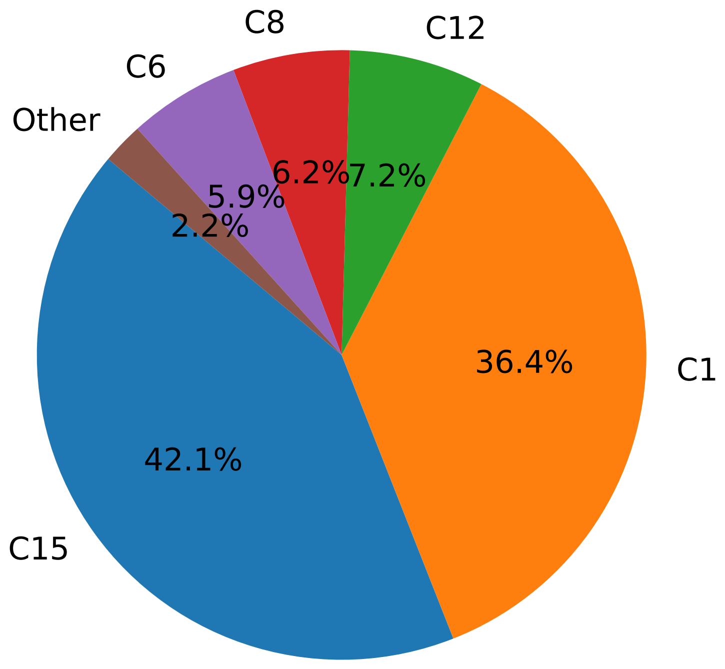

Figure 8. A piechart of Estonian training set cluster proportions. The 'Other' cluster contains eleven smallest clusters.

Appendix C

This appendix displays the t-SNE two-component projections of test sets that were not displayed in the main part of the thesis. These test sets are Estonian Europarl, Estonian Wiki, Estonian Opensubs and English Wiki.

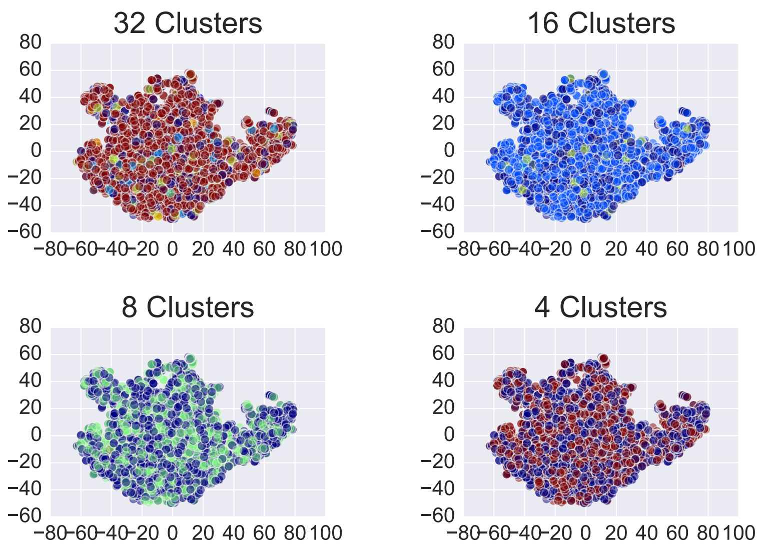

Figure 9. The TSNE 2-component projections of English Wikipedia sentence vectors.

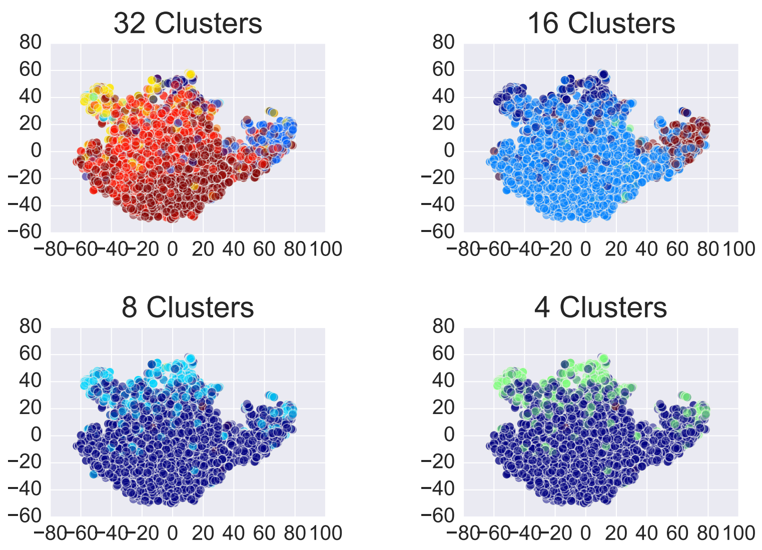

Figure 10. The TSNE 2-component projections of Estonian Opensubs sentence vectors.

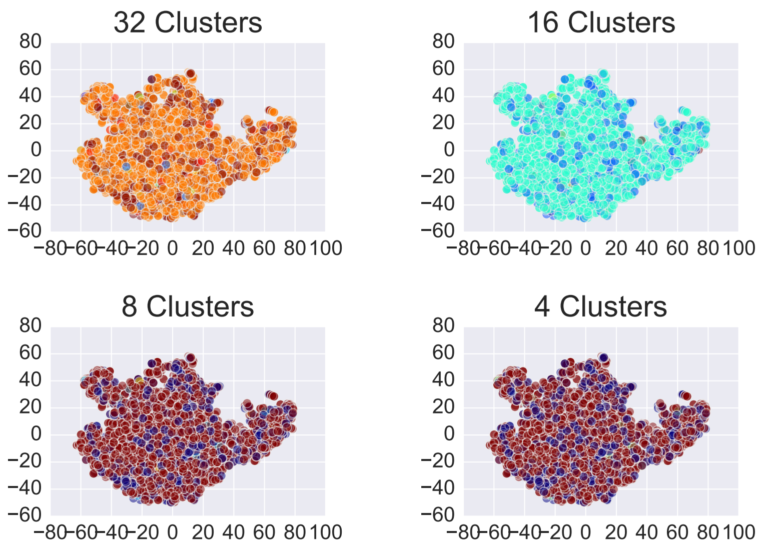

Figure 11. The TSNE 2-component projections of Estonian Europarl sentence vectors.

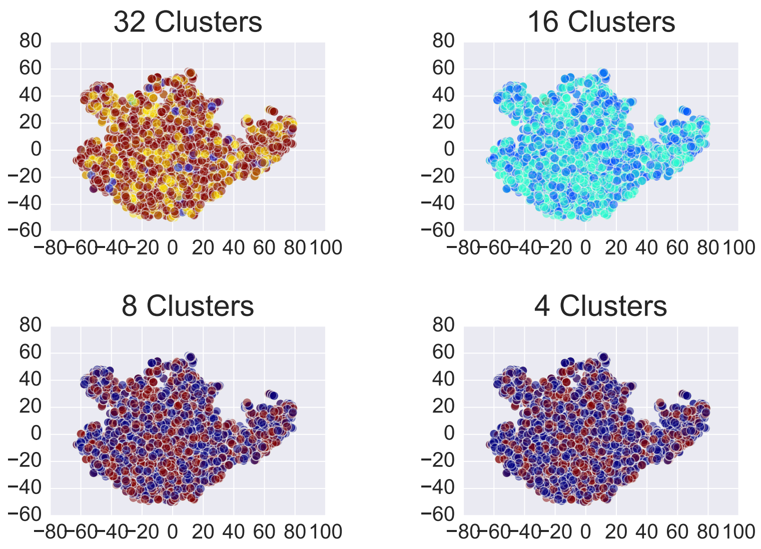

Figure 12. The TSNE 2-component projections of Estonian Wikipedia sentence vectors.

Licence

Non-exclusive licence to reproduce thesis and make thesis public