Deep learning based Semantic Textual Similarity for Applications in Translation Technology

Tharindu Ranasinghe

A thesis submitted in partial fulfilment of the requirements of the University of Wolverhampton for the degree of Doctor of Philosophy

2021

Signature: . . . .

Date: . . . .

Abstract

Acknowledgements

Contents

Abstract **ii**

Acknowledgements **iii**

List of Tables **viii**

List of Figures **xi**

List of Code Listings **xiii**

Introduction **xiv**

I Semantic Textual Similarity **1**

1 Introduction **2**

1.1 Semantic Textual Similarity Approaches 2

1.2 Datasets 4

1.2.1 English Datasets 4

1.2.2 Datasets on Other Languages 19

1.2.3 Datasets on Different Domains 24

1.3 Evaluation Metrics 28

1.4 Contributions 33

2 Improving State of the Art Methods **35**

2.1 Related Work 37

2.1.1 Cosine Similarity on Average Vectors 38

2.1.2 Word Mover's Distance 38

2.1.3 Cosine Similarity Using Smooth Inverse Frequency . . . 38
2.2 Improving State of the Art STS Methods 42
2.3 Portability to Other Languages 52
2.4 Portability to Other Domains 54
2.5 Conclusions 55
3 Exploring STS with Sentence Encoders 58
3.1 Introduction 58
3.2 Related Work 61
3.3 Exploring Sentence Encoders in English STS 66
3.4 Portability to Other Languages 69
3.5 Portability to Other Domains 71
3.6 Conclusions 74
4 Siamese Neural Networks for STS 76
4.1 Introduction 76
4.2 Related Work 79
4.3 Exploring Siamese Neural Networks for STS 81
4.3.1 Impact of Transfer Learning 85
4.3.2 Impact of Data Augmentation 88
4.4 Portability to Other Languages 90
4.5 Portability to Other Domains 93
4.6 Conclusions 95
5 Adopting Transformers for STS 98
5.1 Introduction 98

5.2 Related Work 99

5.3 Transformer Architecture for STS 102

5.4 Exploring Transformers in English STS 102

5.4.1 Impact of Transfer Learning 103

5.4.2 Impact of Data Augmentation 103

5.5 Portability to Other Languages 103

5.6 Potability to Other Domains 103

5.7 Recent Developments: Siamese Transformers 103

5.8 Conclusions 103

II Applications - Translation Memories 104

1 Introduction 105

1.1 What is Translation Memory? 105

1.2 Datasets 105

1.3 Related Work 105

1.4 STS for Translation Memories 105

2 Sentence Encoders for Translation Memories 106

2.1 Introduction 106

2.2 Methodology 106

2.3 Results and Evaluation 106

III Applications - Translation Quality Estimation 107

1 Introduction 108

1.1 What is Translation Quality Estimation? 108
1.2 Datasets 108
1.3 Related Work 108
1.4 STS for Translation Quality Estimation 108
1.5 Conclusion 108
2 TransQuest: STS Architectures for QE 109
2.1 Introduction 109
2.2 Methodology 109
2.3 Results and Evaluation 109
2.4 Conclusion 109
3 Extending TransQuest for word-level QE 112
3.1 Introduction 112
3.2 Related Work 112
3.3 Methodology 112
3.4 Results and Evaluation 112
3.5 Conclusion 112
4 Multilingual Quality Estimation with TransQuest 114
4.1 Introduction 114
4.2 Multilingual Sentence-Level QE 115
4.3 Multilingual Word-Level QE 115
4.4 Conclusion 115
Bibliography 116

List of Tables

I Semantic Textual Similarity 1

1.1 Example sentence pairs from the SICK dataset 6
1.2 Word count stats in SICK 7
1.3 Information about English STS 2017 training set 13
1.4 Example sentence pairs from the STS2017 English dataset 14
1.5 Word count stats in STS 2017 15
1.6 Example question pairs from the Quora Question Pairs dataset . 16
1.7 Word count stats in QUORA 17
1.8 Information about Spanish STS training set 20
1.9 Example sentence pairs from the Spanish STS dataset 22
1.10 Word count stats in Spanish STS 23
1.11 Information about Arabic STS training set 24
1.12 Example question pairs from the Arabic STS dataset 26
1.13 Word count stats in Arabic STS 27
1.14 Example question pairs from the BIOSSES dataset 30

2.1 Results for SICK with Vector Averaging 45
2.2 Results for STS 2017 with Vector Averaging 46
2.3 Results for QUORA with Vector Averaging 46
2.4 Results for SICK with Word Mover's Distance 48

2.5 Results for STS 2017 with Word Mover's Distance 48
2.6 Results for QUORA with Word Mover's Distance 49
2.7 Results for SICK with Smooth Inverse Frequency 49
2.8 Results for STS 2017 with Smooth Inverse Frequency 50
2.9 Results for QUORA with Smooth Inverse Frequency 50

3.1 Results for SICK with sentence encoders 67
3.2 Results for STS 2017 with sentence encoders 68
3.3 Results for QUORA with sentence encoders 68
3.4 Results for Arabic and Spanish STS with Sentence Encoders . . 71
3.5 Results comparison for BIOSSES with different sentence encoders 72

4.1 Results for SICK with Siamese Neural Network 83
4.2 Results for STS 2017 with Siamese Neural Network 84
4.3 Results for QUORA with Siamese Neural Network 84
4.4 Results for transfer learning with Siamese Neural Network . . . 86
4.5 Results for data augmentation with Siamese Neural Network . . 89
4.6 Results comparison for SICK with leader board results 89
4.7 Results comparison for STS2017 with leader board results 90
4.8 Results for Arabic STS with Siamese Neural Network 91
4.9 Results for Spanish STS with Siamese Neural Network 91
4.10 Results comparison for Arabic STS with leader board results . . 92
4.11 Results comparison for Spanish STS with leader board results . 92

4.12 Results for transfer learning with Siamese Neural Network in BIOSSES dataset 94
4.13 Results comparison for BIOSSES with top results 95

5.1 Results for SICK with Transformer Models 102

II Applications - Translation Memories 104

III Applications - Translation Quality Estimation 107

2.1 Pearson correlation between TransQuest algorithm predictions and human post-editing effort 111

3.1 Target F1-Multi between the algorithm predictions and human annotations 113

List of Figures

I Semantic Textual Similarity **1**

1.1 Relatedness distribution of SICK train and SICK test 7

1.2 Normalised distribution of word count in SICK train and SICK test. 8

1.3 Word share against relatedness bins in SICK train and SICK test. 9

1.4 Relatedness distribution of STS 2017 train and STS 2017 test . . 11

1.5 Normalised distribution of word count in STS 2017 train and STS 2017 test. 12

1.6 Word share against relatedness bins in STS 2017 train and STS 2017 test. 12

1.7 Is-duplicate distribution of QUORA train and QUORA test . . . 17

1.8 Normalised distribution of word count in QUORA train and QUORA test. 17

1.9 Word share against Is-duplicate values in QUORA train and QUORA test. 18

1.10 Relatedness distribution of Spanish STS train and Spanish STS test 20

1.11 Normalised distribution of word count inSpanish STS train and Spanish STS test. 21

1.12 Word share against relatedness bins in Spanish STS train and STS 2017 test. 21

1.13 Relatedness distribution of Arabic STS train and Arabic STS test 25
1.14 Normalised distribution of word count in Arabic STS train and Arabic STS test. 25
1.15 Word share against relatedness bins in Arabic STS train and Arabic STS test. 27
1.16 Relatedness distribution of BIOSSES 29
1.17 Normalised distribution of word count in BIOSSES. 29
1.18 Word share against relatedness bins in BIOSSES. 31

2.1 The Word Mover's Distance between two sentences 39

3.1 Infersent Architecture 63
3.2 Universal Sentence Encoder Architectures 65

4.1 Basic structure of the Siamese neural network 82

5.1 Basic architecture for using Transformers in STS 102

II Applications - Translation Memories 104

III Applications - Translation Quality Estimation 107

2.1 Architecture in TransQuest 110

Listings

Abstract

Semantic textual similarity (STS) is a natural language processing (NLP) task to quantitatively assess the semantic similarity between two text snippets. STS is a fundamental NLP task for many text-related applications, including text deduplication, paraphrase detection, semantic searching, and question answering. Measuring STS is a machine learning (ML) problem, where a ML model predicts a value that represents the similarity of the two input texts. These machine learning models can be categorised in to two main areas; supervised and unsupervised. Supervised STS ML models have been trained on an annotated STS dataset while the unsupervised STS ML models predict STS without being trained on annotated STS data. In the first part of the thesis we explore supervised and unsupervised ML models in STS. We explore embedding aggregation based state-of-the-art STS methods, sentence encoders, Siamese neural networks and transformers in STS. Furthermore, for each STS method we analyse the ability of the model to perform in a multilingual and multi-domain setting. On the process, we develop new state-of-the-art unsupervised STS method based on contextual word embeddings and new state-of-the-art supervised STS method based on Siamese neural networks.

The second and third parts of the thesis, focus on applying the developed STS method in the applications of translation technology; translation memories

(TM) and translation quality estimation (QE). We identify that the edit distance based matching and retrieval algorithms in TMs are less efficient and we propose a TM matching algorithm based on the STS methods we developed in the first part of the thesis. We empirically show that this algorithm outperforms edit distance based matching algorithms. As the next application, we utilise the STS architectures we developed, in translation quality estimation. We show that STS architectures can be successfully applied in QE by changing the input embeddings in to cross-lingual embeddings. Based on that , we develop TransQuest - a new state-of-the-art QE framework that won the WMT 2020 QE shared task. We release TransQuest as an open-source library and by the time of writing this, TransQuest has more than 8,000 downloads from the community.

Part I

Semantic Textual Similarity

Chapter 1

Introduction

1.1 Semantic Textual Similarity Approaches

Over the years, researchers have proposed numerous STS methods. Most of the early approaches were based on traditional machine learning and involved heavy feature engineering [1]. With the advances of word embeddings, and as a result of the success neural networks have achieved in other fields, most of the methods proposed in recent years rely on neural architectures [2, 3]. Neural networks are preferred over traditional machine learning models as they generally tend to perform better than traditional machine learning models. They also do not rely on explicit linguistics features which have to be extracted before the ML model is learnt. Determining the best linguistic features for calculating STS is not an easy task as it requires a good understanding of the linguistic phenomenon and relies on researchers' intuition. In addition, calculating these features is usually not an easy task, especially for languages other than English. Therefore, in contrast to traditional ML methods, models based on word embeddings and neural networks can be easily applied to other languages.

As stated in the Chapter the machine learning algorithms we experimented can be classified in to two main categories: Unsupervised STS methods and Supervised STS methods. In the Chapter 2 we evaluate the current STS state of the arts methods that uses word embeddings and we improve state of the arts STS methods using contextual embeddings.

In Chapter 3 we explore another unsupervised STS method using sentence encoders. We use three different sentence encoders and analyse their performance in various aspects of English STS and also evaluate their portability to different languages and domains.

Siamese Neural Networks are a special kind of neural network that are being used commonly in STS tasks. It is a supervised STS method which we discuss comprehensively in Chapter 4. We evaluate the existing Siamese Neural Network architectures in STS datasets and propose a novel Siamese Neural Network architecture, MAGRU: an efficient and more accurate Siamese Neural Network architecture for STS tasks. We also asses its performance on different languages and different domains.

In the final chapter of the Part I of this thesis, we explore the newly released transformers in STS tasks. We bring together various transformer architectures like BERT [4], XLNet [5], RoBERTa [6] etc and investigate their performance in various STS datasets in Chapter 5.

The remainder of this chapter is structured as follows. Section 1.2 discuss the various datasets we used in *"Semantic Textual Similarity"* part of the thesis. We also briefly analyse the datasets for common properties. In the Section 1.4 we discuss the main contributions we have to the community with the *"Semantic Textual Similarity"* part of the thesis. The chapter concludes with the conclusions.

1.2 Datasets

We experimented with several datasets throughout the experiments in the Semantic Textual Similarity Section. In order to maintain the versatility of our methods we experimented with several English datasets as well as several non English datasets and a dataset from a different domain which we will introduce in this section. All of the datasets which are described here are publicly available and can be considered as STS benchmarks.

1.2.1 English Datasets

1. **SICK dataset** [1] - The SICK data contains 9927 sentence pairs with a 5,000/4,927 training/test split which were employed in the SemEval 2014 Task1: Evaluation of Compositional Distributional Semantic Models on Full Sentences through Semantic Relatedness and Textual Entailment [7]. The dataset has two types of annotations: Semantic Relatedness and Textual Entailment. We only use Semantic Relatedness annotations in our research. SICK was built starting from two existing datasets: the 8K ImageFlickr data set [2] [8] and the SemEval-2012 STS MSR-Video Descriptions dataset [3] [9]. The 8K ImageFlickr dataset is a dataset of images, where each image is associated with five descriptions. To derive SICK sentence pairs the organisers randomly selected 750 images and sampled two descriptions from each of

[1]The SICK dataset is available to download at https://wiki.cimec.unitn.it/tiki-index.php?page=CLIC
[2]The 8K ImageFlickr data set is available at http://hockenmaier.cs.illinois.edu/8k-pictures.html
[3]The SemEval-2012 STS MSR-Video Descriptions dataset is available at https://www.cs.york.ac.uk/semeval-2012/task6/index.html

them. The SemEval2012 STS MSR-Video Descriptions data set is a collection of sentence pairs sampled from the short video snippets which compose the Microsoft Research Video Description Corpus [4]. A subset of 750 sentence pairs have been randomly chosen from this data set to be used in SICK.

In order to generate SICK data from the 1,500 sentence pairs taken from the source data sets, a 3-step process has been applied to each sentence composing the pair, namely *(i) normalisation, (ii) expansion and (iii) pairing* [7]. The *normalisation* step has been carried out on the original sentences to exclude or simplify instances that contained lexical, syntactic or semantic phenomena such as named entities, dates, numbers, multiword expressions etc. In the *expansion* step syntactic and lexical transformations with predictable effects have been applied to each normalized sentence, in order to obtain *(i)* a sentence with a similar meaning, *(ii)* a sentence with a logically contradictory or at least highly contrasting meaning, and *(iii)* a sentence that contains most of the same lexical items, but has a different meaning. Finally, in the *pairing* step each normalised sentence in the pair has been combined with all the sentences resulting from the expansion phase and with the other normalised sentence in the pair. Furthermore, a number of pairs composed of completely unrelated sentences have been added to the data set by randomly taking two sentences from two different pairs [7].

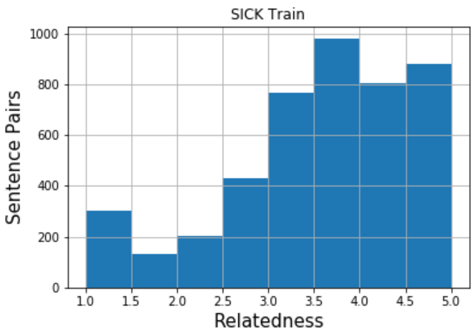
[4] The Microsoft Research Video Description Corpus is available to download at https://research.microsoft.com/en-us/downloads/38cf15fd-b8df-477e-a4e4-a4680caa75af/

Each pair in the SICK dataset has been annotated to mark the degree to which the two sentence meanings are related (on a 5-point scale). The ratings have been collected through a large crowdsourcing study, where each pair has been evaluated by 10 different annotators. Once all the annotations were collected, the relatedness gold score has been computed for each pair as the average of the ten ratings assigned by the annotators [7]. Table 1.1 shows examples of sentence pairs with different degrees of semantic relatedness; gold relatedness scores are expressed on a 5-point rating scale. Given a test sentence pair the machine learning models require to predict a value between 0-5 which reflects the relatedness of the given sentence pair.

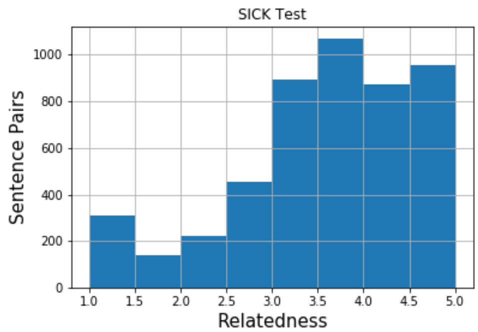
Sentence Pair	**Relatedness**
1. A little girl is looking at a woman in costume. 2. A young girl is looking at a woman in costume.	4.7
1. Nobody is pouring ingredients into a pot. 2. Someone is pouring ingredients into a pot.	3.5
1. Someone is pouring ingredients into a pot. 2. A man is removing vegetables from a pot.	2.8
1. A man is jumping into an empty pool. 2. There is no biker jumping in the air.	1.6

Table 1.1: Example sentence pairs from the SICK dataset with their gold relatedness scores (on a 5-point rating scale). **Sentence Pair** column shows the two sentence and **Relatedness** column denotes the annotated relatedness score.

Figure 1.1 shows the distribution of the relatedness value in SICK training and SICK testing set. It is clear that there are more sentence pairs with a high relatedness values compared to low relatedness values. SICK train



(a) Relatedness distribution of SICK training test



(b) Relatedness distribution of SICK testing set

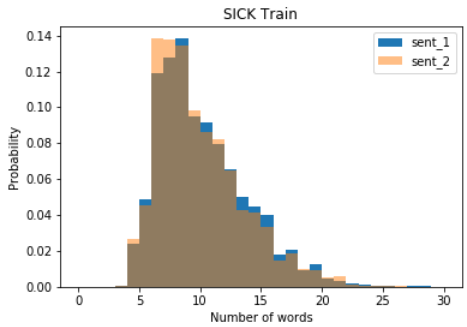
Figure 1.1: Relatedness distribution of SICK train and SICK test. *Sentence Pairs* shows the number of sentence pairs that a certain *Relatedness bin* has.

Measure	SICK Train		SICK Test	
	Sent_1	**Sent_2**	**Sent_1**	**Sent_2**
Word Count Mean	9.73	9.52	9.69	9.53
Word Count STD	3.66	3.70	3.69	3.65
Word Count MAX	28	32	28	30
Word Count MIN	3	3	3	3

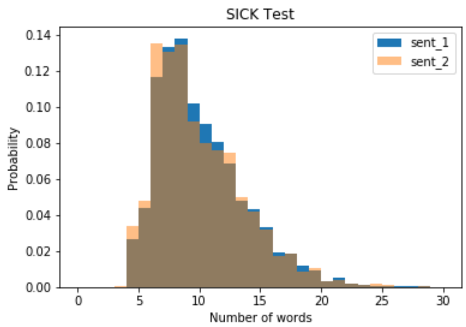
Table 1.2: Word count stats in SICK training and SICK testing. *STD* indicates the standard deviation and the other acronyms indicate the common meaning

and SICK test follows a similar distribution.

In Figure 1.2 we visualise the normalised distribution of word count for both sentence 1 and sentence 2 in SICK train and SICK test. Both sentences have a similar distribution reaching the maximum around 9 words. SICK train and SICK test follows a similar pattern in word count distribution too. Additionally we show some word count statistics in Table 1.2. In SICK train number of words for a sentence ranges from 3 to 32 and have the mean number of words around 9.5. These statistics are extremely close in



(a) Normalised distribution of word count in SICK train



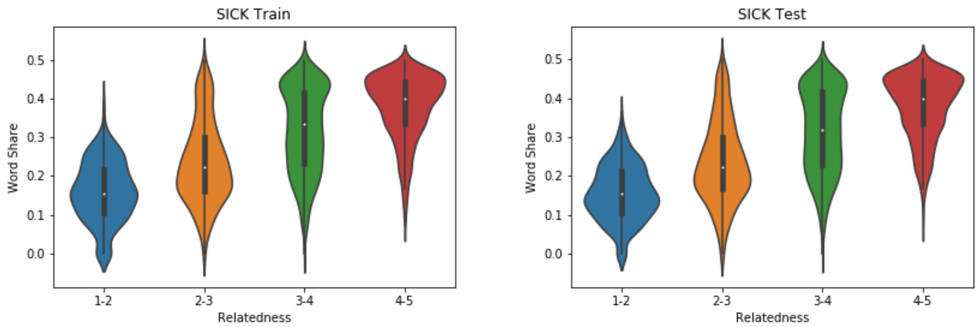
(b) Normalised distribution of word count in SICK test

Figure 1.2: Normalised distribution of word count in SICK train and SICK test. *Number of words* indicates the word count and *Probability* shows the total probability of a sentence with that word count appearing in the dataset.

SICK test too.

The common judgement in STS is that, when two sentences share a large number of words, the relatedness of that two sentences should be higher. In fact, in early feature based approaches of calculating semantic textual similarity, the number of overlapping words between the two sentences was a common feature [10, 11, 12, 13]. Systems like Vilariño et al. [10], Lynum et al. [12] use the number of words common in two sentences as a feature directly while systems like Gupta et al. [11], Chávez et al. [13] use Jaccard Similarity Coefficient as a feature, which is a measurement based on word overlap. To observe, whether the number of words common in the two sentences has a relationship on the relatedness, we draw a violin plot [5] for each relatedness score bins with word share in Figure 1.3.

[5]Violin plots are similar to box plots, except that they also show the probability density of the data at different values, usually smoothed by a kernel density estimator.



(a) Word share against relatedness bins in SICK train

(b) Word share against relatedness bins in SICK test

Figure 1.3: Word share against relatedness bins in SICK train and SICK test. *Word Share* indicates the ratio between number of common words in the two sentences to total number of words in the two sentences against each *Relatedness* bins

In figure 1.3, it is clear that sentence pairs with a higher relatedness tend to have a high word share. However, it should be noted that, in the "2-3" relatedness score bin, there are some sentence pairs with a high word share. Most common example for such a case would be sentence 2 is the complete negation of the sentence 1. In such cases the two sentences share a large potion of the words and one sentence have the *"not"* word that gives a complete opposite meaning compared to the other sentence. Similarly "4-5" relatedness score bin has some sentence pairs with a low word share. Those sentence pairs does not contain the same words but will be having synonyms and possess the same overall meaning. Therefore, the STS methods that focusses on word share won't perform well in SICK dataset.

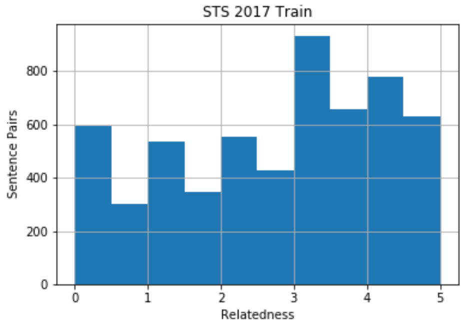
A clear strength in the SICK dataset is that training set and the testing set reflects similar properties so that a properly trained machine learning model on SICK train should give good results to the SICK test set as well.

2. **STS 2017 English Dataset** [6] The second English STS dataset we used to experiment in this section is STS 2017 English Dataset which was employed in SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Cross-lingual Focused Evaluation which is the most recent STS task in SemEval [14]. As the training data for the competition, participants were encouraged to make use of all existing data sets from prior STS evaluations including all previously released trial, training and evaluation data from SemEval 2012 - 2016 [9, 15, 16, 17, 18]. Once combined we had 8277 sentence pairs for training. More information about the datasets used to build the training set is available in Table 1.3.

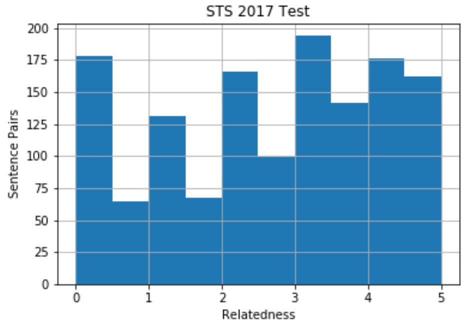
   On the other hand, a fresh test set of 250 sentence pairs was provided by SemEval-2017 STS Task organisers [14]. The Stanford Natural Language Inference (SNLI) corpus [19] was the primary data source for this test set. Similar to the SICK dataset, Each pair in the STS 2017 English Test set has been annotated to mark the degree to which the two sentence meanings are related (on a 5-point scale). The ratings have been collected through crowdsourcing on Amazon Mechanical Turk[7]. Five annotations have been collected per pair and gold score has been computed for each pair as the average of the five ratings assigned by the annotators. However, unlike the

[6] The STS 2017 English Dataset is available to download at http://ixa2.si.ehu.es/stswiki/

[7] Amazon Mechanical Turk is a crowdsourcing website for businesses to hire remotely located *crowd workers* to perform discrete on-demand tasks. It is available at https://www.mturk.com/



(a) Relatedness distribution of STS 2017 training test

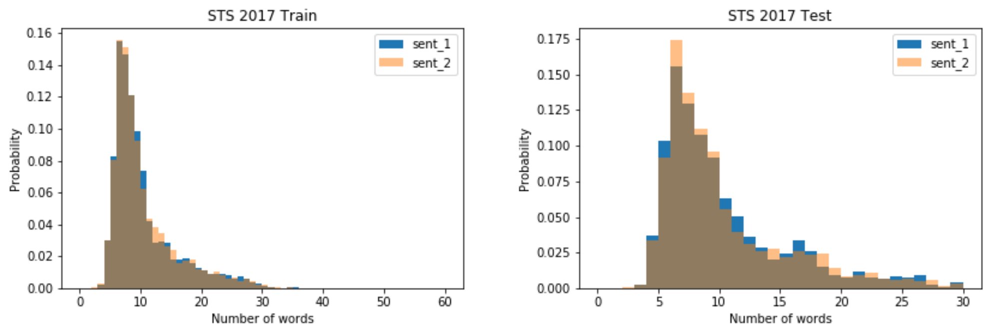


(b) Relatedness distribution of STS 2017 testing set

Figure 1.4: Relatedness distribution of STS 2017 train and STS 2017 test. *Sentence Pairs* shows the number of sentence pairs that a certain *Relatedness bin* has.

SICK dataset, the organisers has a clear explanations for the score ranges. Table 1.4 shows some example sentence pairs from the dataset with the gold labels and their explanations. Similar to the SICK dataset, the machine learning models require to predict a value between 0-5 which reflects the similarity of the given sentence pair.

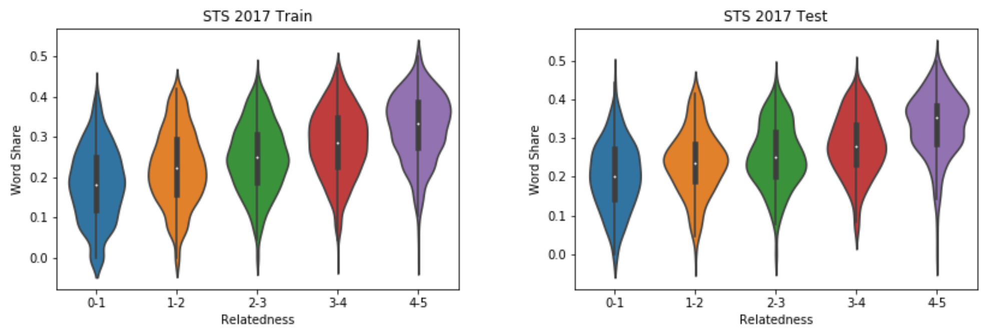
Similar to the SICK dataset, we calculate some statistics and produce some graphs. Figure 1.4 shows the relatedness distribution and Figure 1.5 shows the normalised distribution of word count for sentence 1 and sentence 2 in STS 2017 train and test sets. Most of these statistics are similar to the SICK dataset. One notable change is the maximum word count in STS 2017 training dataset which is 57 in sentence 1 and 48 in sentence 2 according to Table 1.5 while both SICK datasets' and STS 2017 test set's maximum word count is limited to 30. We believe that the reason is STS train is composed with many sources including news articles which can have lengthy



(a) Normalised distribution of word count in STS 2017 train

(b) Normalised distribution of word count in STS 2017 test

Figure 1.5: Normalised distribution of word count in STS 2017 train and STS 2017 test. *Number of words* indicates the word count and *Probability* shows the total probability of a sentence with that word count appearing in the dataset.



(a) Word share against relatedness bins in STS 2017 train

(b) Word share against relatedness bins in STS 2017 test

Figure 1.6: Word share against relatedness bins in STS 2017 train and STS 2017 test. *Word Share* indicates the ratio between number of common words in the two sentences to total number of words in the two sentences against each *Relatedness* bins

Year	Dataset	Pairs	Source
2012 [9]	MSRpar	1500	newswire
	MSRvid	1500	videos
	OnWN	750	glosses
	SMTnews	750	WMT eval.
	SMTeuroparl	750	WMT eval.
2013 [15]	HDL	750	newswire
	FNWN	189	glosses
	OnWN	561	glosses
	SMT	750	MT eval.
2014 [16]	HDL	750	newswire headlines
	OnWN	750	glosses
	Deft-forum	450	forum posts
	Deft-news	300	news summary
	Images	750	image descriptions
	Tweet-news	750	tweet-news pairs
2015 [17]	HDL	750	newswire headlines
	Images	750	image descriptions
	Ans.-student	750	student answers
	Ans.-forum	375	Q&A forum answers
	Belief	375	committed belief
2016 [18]	HDL	249	newswire headlines
	Plagiarism	230	short-answer plag.
	post-editing	244	MT postedits
	Ans.-Ans.	254	Q&A forum answers
	Quest.-Quest.	209	Q&A forum questions
2017 [14]	Trial	23	Mixed STS 2016

Table 1.3: Information about the datasets used to build the English STS 2017 training set. The **Year** column shows the year of the SemEval competition that the dataset got released. **Dataset** column expresses the acronym used describe a dataset in that year. **Pairs** is the number of sentence pairs in that particular dataset and **Source** shows the source of the sentence pairs.

sentences. However, the STS algorithm should be able to properly handle this imbalance nature between STS 2017 train and test set.

In Figure 1.6 we draw a violin plot for each relatedness score bin with

Sentence Pair	Relatedness
The two sentences are completely equivalent as they mean the same thing. 1. The bird is bathing in the sink. 2. Birdie is washing itself in the water basin.	5
The two sentences are completely equivalent as they mean the same thing. 1. The bird is bathing in the sink. 2. Birdie is washing itself in the water basin.	4
The two sentences are roughly equivalent, but some important information differs/missing. 1. John said he is considered a witness but not a suspect. 2. "He is not a suspect anymore." John said.	3
The two sentences are not equivalent, but share some details. 1. They flew out of the nest in groups. 2. They flew into the nest together.	2
The two sentences are not equivalent, but are on the same topic. 1. The woman is playing the violin. 2. The young lady enjoys listening to the guitar.	1
The two sentences are completely dissimilar 1. The black dog is running through the snow. 2. A race car driver is driving his car through the mud.	0

Table 1.4: Example sentence pairs from the STS2017 English dataset with their gold relatedness scores (on a 5-point rating scale) and explanations. **Sentence Pair** column shows the two sentence and **Relatedness** column denotes the annotated relatedness score.

word share. We can see that generally higher word share leads to higher relatedness, but still there can be sentence pairs contradicts this which is similar to the observation we had with SICK dataset.

Since the statics of SICK and STS 2017 datasets are similar one dataset can be used to augment the training data in the other dataset which can lead to

Measure	STS 2017 Train		STS 2017 Test	
	Sent_1	**Sent_2**	**Sent_1**	**Sent_2**
Word Count Mean	10.01	9.94	9.83	9.80
Word Count STD	5.52	5.36	5.14	5.14
Word Count MAX	57	48	30	30
Word Count MIN	3	2	3	2

Table 1.5: Word count stats in STS 2017 training and STS 2017 testing.*STD* indicates the standard deviation and the other acronyms indicate the common meaning

better results as neural networks perform better with more data [20, 21]. We hope to experiment this with supervised machine learning models in Chapters 4 and 5.

3. **Quora Question Pairs** [8] The Quora Question Pairs dataset is a big dataset which was first released for a Kaggle Competition[9]. Quora is a question-and-answer website where questions are asked, answered, followed, and edited by internet users, either factually or in the form of opinions. If a particular new question has been asked before, users merge the new question to the original question flagging it as a duplicate. The organisers used this functionality to create the dataset and did not use a separate annotation process. Their original sampling method has returned an imbalanced dataset with many more true examples of duplicate pairs than non-duplicates. Therefore, the organisers have supplemented the dataset with negative examples. One source of negative examples have been pairs of

[8]The Quora Question Pairs Dataset is available to download at `http://qim.fs.quoracdn.net/quora_duplicate_questions.tsv`

[9]Kaggle is an online community of data scientists and machine learning practitioners that hosts machine learning competitions. The Quora Question Pairs competition is available on `https://www.kaggle.com/c/quora-question-pairs`

related question which, although pertaining to similar topics, are not truly semantically equivalent.

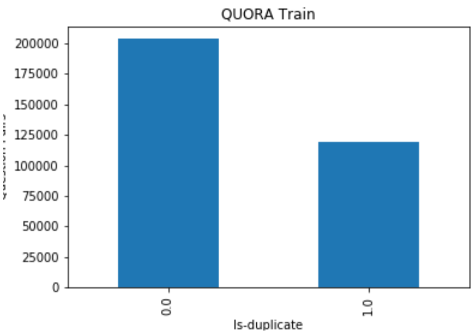
The dataset has 400,000 question pairs and we used 4:1 split on that to separate it into a training set and a test set resulting 320,000 questions pairs in the training set and 80,000 sentence pairs in the testing set. The machine learning models need to predict a value between 0 and 1 that reflects whether it is a duplicate question pair or not. 1 indicates that a certain question pair is a duplicate and 0 indicates it is not a duplicate.

Question Pair	**is-duplicate**
1. What are natural numbers? 2. What is a least natural number?	0
1. Which Pizzas are most popularly ordered in Dominos menu? 2. How many calories does a Dominos Pizza have?	0
1. How do you start a bakery? 2. How can one start a bakery business?	1
1. Should I learn Python or Java first? 2. If I had to choose between learning Java and Python what should I choose to learn first?	1

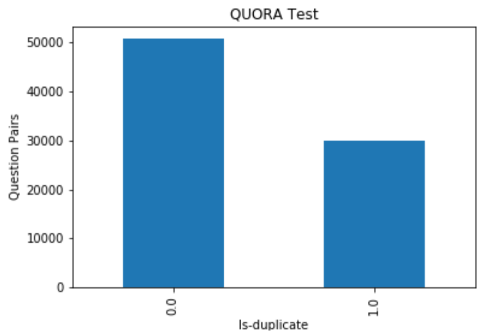
Table 1.6: Example question pairs from the Quora Question Pairs dataset with their gold is-duplicate value. **Question Pair** column shows the two questions and **is-duplicated** column denotes whether it is a duplicated pair or not.

This is different to the previous datasets since it is not artificially created and use day to day language. Since it has more than 300,000 training instances deep learning systems will benefit more when used on this dataset.

In Figure 1.7 we show the distribution of the two classes in QUORA dataset.

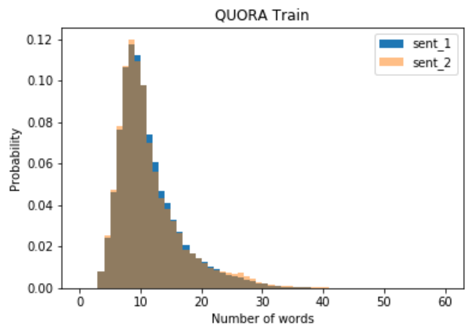


(a) Is-duplicate distribution of QUORA training test

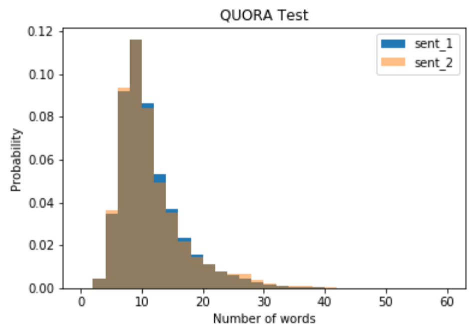


(b) Is-duplicate distribution of QUORA testing set

Figure 1.7: Is-duplicate distribution of QUORA train and QUORA test. *Sentence Pairs* shows the number of sentence pairs that a certain *Is-duplicate* has.



(a) Normalised distribution of word count in QUORA train

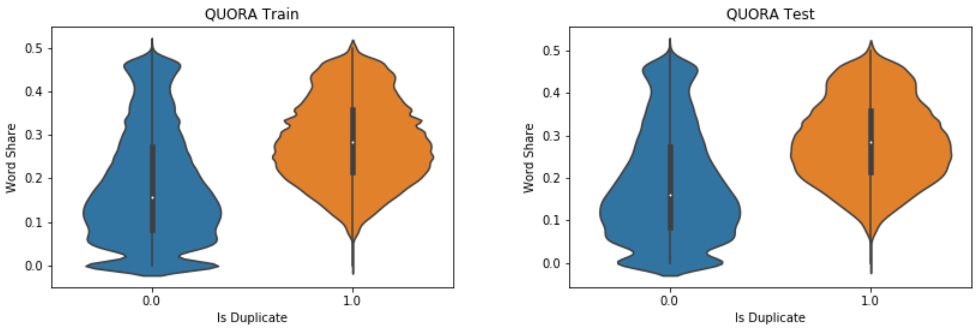


(b) Normalised distribution of word count in QUORA test

Figure 1.8: Normalised distribution of word count in QUORA train and QUORA test. *Number of words* indicates the word count and *Probability* shows the total probability of a sentence with that word count appearing in the dataset.

Measure	QUORA Train		QUORA Test	
	Ques_1	**Ques_2**	**Ques_1**	**Ques_2**
Word Count Mean	10.95	11.20	10.92	11.14
Word Count STD	5.44	6.31	5.40	6.31
Word Count MAX	125	237	73	237
Word Count MIN	1	1	1	1

Table 1.7: Word count stats in QUORA training and QUORA testing.*STD* indicates the standard deviation and the other acronyms indicate the common meaning



(a) Word share against is-relatedness value in QUORA train (b) Word share against is-relatedness value in QUORA test

Figure 1.9: Word share against Is-duplicate values in QUORA train and QUORA test. *Word Share* indicates the ratio between number of common words in the two sentences to total number of words in the two sentences against each *Is-duplicate*

The dataset seems to have more non duplicate question pairs than duplicate sentence pairs which is similar to the real world scenario. According to the word count distribution in Figure 1.8 and word count statistics in Table 1.7, it is clear that QUORA datasets contains longer texts than SICK and STS 2017 datasets. Therefore, QUORA dataset should be able to test machine learning models'ability to handle lengthy texts properly.

In Figure 1.9 we show a violin plot for each *"is-duplicate"* value with word share. We can see that duplicate questions have a high word share. However, it should be noted that there are non duplicate question pairs that still have a high word share. The machine learning algorithm should be able to handle them properly.

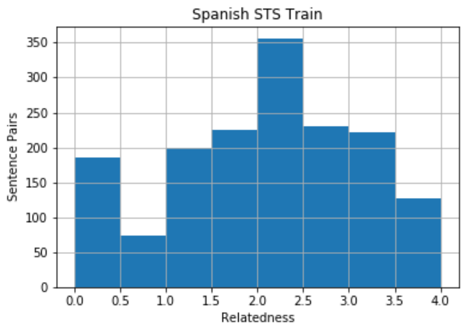
According to statistics provided by the Director of Product Management at Quora on 17 September 2018, over 100 million people visit Quora every month, which raises the problem of different users asking similar questions with same intent but in different words [22]. Multiple questions with the same intent can cause seekers to spend more time finding the best answer to their question, and make writers feel they need to answer multiple versions of the same question. Therefore, identifying duplicate questions will make it easier to find high quality answers to questions resulting in an improved experience for Quora writers, seekers, and readers.

1.2.2 Datasets on Other Languages

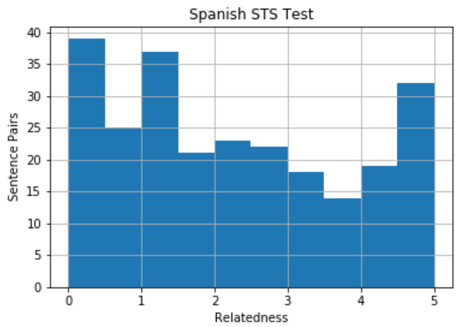
One of the main requirements in our research was to build a STS method without depending on the language. Therefore through out our research we worked on several datasets from different languages. Those non-English datasets are described below.

1. **Spanish STS Dataset** [10] - Spanish STS dataset that we used was employed for Spanish STS subtask in SemEval 2017 Task 1: Semantic Textual Similarity Multilingual and Cross-lingual Focused Evaluation [14]. The training set has 1250 sentence pairs annotated with a relatedness score between 0 and 4. The training set combined several datasets from previous SemEval STS shared tasks also[14]. Table 1.8 shows more information about the training set. There were two sources for test set - Spanish news and Spanish Wikipedia dump having 500 and 250 sentence pairs respectively [14]. Both datasets were annotated with a relatedness score between 0 and 5.

[10] The Spanish STS dataset can be downloaded at http://alt.qcri.org/semeval2017/task1/index.php?id=data-and-tools



(a) Relatedness distribution of Spanish STS training test



(b) Relatedness distribution of Spanish STS testing set

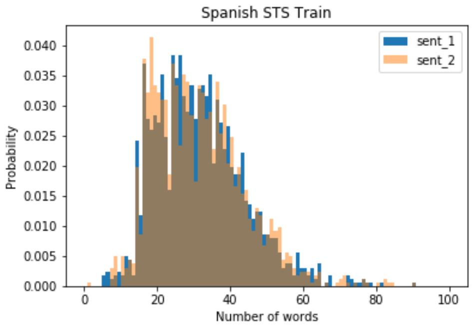
Figure 1.10: Relatedness distribution of Spanish STS train and Spanish STS test. *Sentence Pairs* shows the number of sentence pairs that a certain *Relatedness bin* has.

Table 1.9 shows few pairs of sentences with their similarity score. The machine learning models require to predict a value between 0-5 which reflects the similarity of the given Spanish sentence pair.

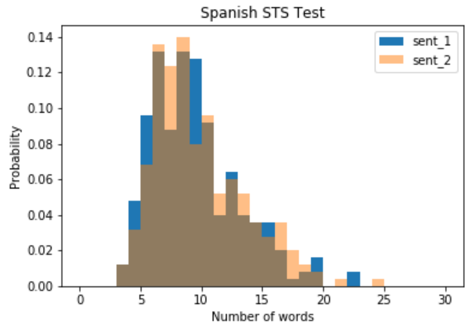
Year	Dataset	Pairs	Source
2014 [16]	Trial	56	NR
	Wiki	324	Spanish Wikipedia
	News	480	Newswire
2015 [16]	Wiki	251	Spanish Wikipedia
	News	500	Sewswire

Table 1.8: Information about the datasets used to build the Spanish STS training set. The **Year** column shows the year of the SemEval competition that the dataset got released. **Dataset** column expresses the acronym used describe a dataset in that year. **Pairs** is the number of sentence pairs in that particular dataset and **Source** shows the source of the sentence pairs.

Similar to the English datasets we calculate some statistics and produce some graphs. A key challenge in the Spanish STS dataset is that test set



(a) Normalised distribution of word count in Spanish STS train



(b) Normalised distribution of word count in Spanish STS test

Figure 1.11: Normalised distribution of word count in Spanish STS train and Spanish STS test. *Number of words* indicates the word count and *Probability* shows the total probability of a sentence with that word count appearing in the dataset.

(a) Word share against relatedness bins in Spanish STS train

(b) Word share against relatedness bins in Spanish STS test

Figure 1.12: Word share against relatedness bins in Spanish STS train and Spanish STS test. *Word Share* indicates the ratio between number of common words in the two sentences to total number of words in the two sentences against each *Relatedness* bins

Sentence Pair	Similarity
1. Amás, los misioneros apunten que los númberos d'infectaos puen ser shasta dos o hasta cuatro veces más grandess que los oficiales. *(Furthermore, missionaries point out that the numbers of infected can be up to two or up to four times larger than the official ones.)* 2. Los cadáveres de personas fallecidas pueden ser hasta diez veces más contagiosos que los infectados vivos. *(The corpses of deceased people can be up to ten times more contagious than those infected alive.)*	0.6
1. La policía abatió a un caníbal cuando devoraba a una mujer Matthew Williams, de 34 años, fue sorprendido en la madrugada mordiendo el rostro de una joven a la que había invitado a su hotel. *(Police killed a cannibal while devouring a woman Matthew Williams, 34, was caught early in the morning biting the face of a young woman he had invited to his hotel.)* 2. La policía de Gales del Sur mató a un caníbal cuando se estaba comiendo la cara de una mujer de 22 años en la habitación de un hotel. *(South Wales police killed a cannibal when he was eating the face of a 22-year-old woman in a hotel room.)*	2
1. Ollanta Humala se reúne mañana con el Papa Francisco. *(Ollanta Humala meets tomorrow with Pope Francis.)* 2. El Papa Francisco mantuvo hoy una audiencia privada con el presidente Ollanta Humala, en el Vaticano. *(Pope Francis held a private audience today with President Ollanta Humala, at the Vatican.)*	3

Table 1.9: Example sentence pairs from the Spanish STS dataset. **Sentence Pair** column shows the two sentences. We also included their translations in the table. The translations were done by a native Spanish speaker. **Similarity** column indicates the annotated similarity of the two sentences.

is very different from the training set. As can be seen in Figure 1.10 training set has been annotated with relatedness scores 0-4 while the test set has been annotated with relatedness scores 0-5. Therefore, STS methods

Measure	Spanish STS Train		Spanish STS Test	
	Sent_1	**Sent_2**	**Sent_1**	**Sent_2**
Word Count Mean	31.23	31.02	9.03	9.34
Word Count STD	12.15	12.37	3.66	3.74
Word Count MAX	90	90	22	24
Word Count MIN	5	1	3	3

Table 1.10: Word count stats in Spanish STS training and Spanish STS testing.*STD* indicates the standard deviation and the other acronyms indicate the common meaning

should be able to handle that properly. Furthermore, as shown in Figure 1.11 and in Table 1.10 sentence pairs in test set are shorter in word length than the sentence pairs in train set. Therefore, STS methods working on this dataset should be able to properly handle that too. This can be observed as a weakness in this dataset, but at the same time this property of the dataset can be exploited to measure the strength of a STS system as well.

2. **Arabic STS Dataset** [11] The Arabic STS dataset we selected was also used for the Arabic STS subtask in SemEval 2017 Task 1: Semantic Textual Similarity Multilingual and Cross-lingual Focused Evaluation [14]. Unlike Spanish, no data from previous SemEval competitions were available since this was the first time an Arabic STS task was organised in SemEval. More information about the extracted sentences will be shown in the Table 1.11.

   To prepare the annotated instances, a subset of the English STS 2017 dataset has been selected and human translated into Arabic. Sentences have been

[11] The Arabic STS dataset can be downloaded at http://alt.qcri.org/semeval2017/task1/index.php?id=data-and-tools

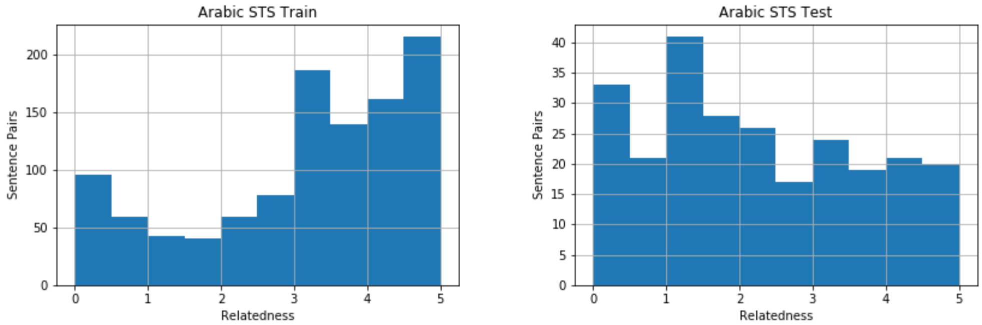
Dataset	Pairs	Source
Trial	23	Mixed STS 2016
MSRpar	510	newswire
MSRvid	368	videos
SMTeuroparl	203	WMT eval.

Table 1.11: Information about the datasets used to build the Arabic STS training set. **Dataset** column expresses the acronym used describe the dataset. **Pairs** is the number of sentence pairs in that particular dataset and **Source** shows the source of the sentence pairs.

translated independently from their pairs. Arabic translation has been provided by native Arabic speakers with strong English skills in Carnegie Mellon University in Qatar. Translators have been given an English sentence and its Arabic machine translation5 where they have performed post-editing to correct errors. STS labels have been then transferred to the translated pairs. Therefore, annotation guidelines and the template will be similar to the English STS 2017 dataset. 1103 sentence pairs were available for training and 250 sentence pairs were available in the test set. Table 1.12 shows few pairs of sentences with their similarity score. The machine learning models require to predict a value between 0-5 which reflects the similarity of a given Arabic sentence pair.

1.2.3 Datasets on Different Domains

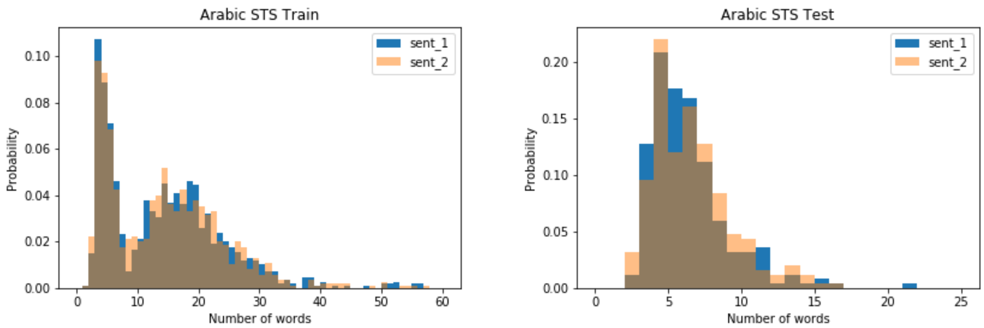
In order to experiment how our STS methods can be adopted in to different domains we also used a dataset from a different discipline which we introduce in this section.



(a) Relatedness distribution of Arabic STS training test

(b) Relatedness distribution of Arabic STS testing set

Figure 1.13: Relatedness distribution of Arabic STS train and Arabic STS test. *Sentence Pairs* shows the number of sentence pairs that a certain *Relatedness bin* has.



(a) Normalised distribution of word count in Spanish STS train

(b) Normalised distribution of word count in Spanish STS test

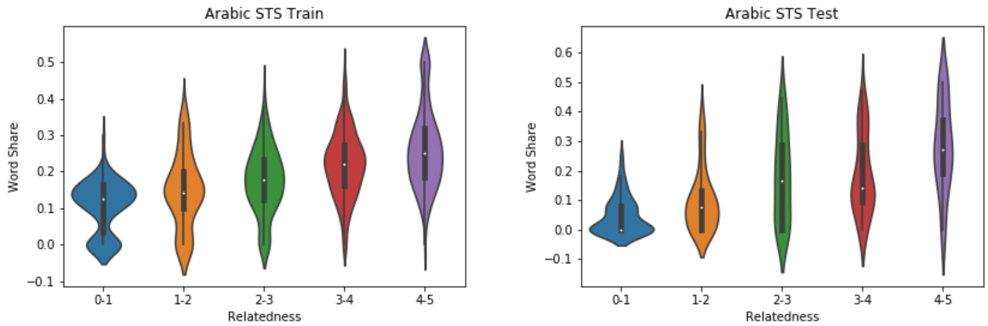
Figure 1.14: Normalised distribution of word count in Arabic STS train and Arabic STS test. *Number of words* indicates the word count and *Probability* shows the total probability of a sentence with that word count appearing in the dataset.

Sentence Pair	Similarity
1. أحدهم يقلي لحما. *Someone is frying meat.* 2. أحدهم يعزف البيانو. *Someone plays the piano.*	0.250
1. أمرأة تظيف المكونات في الإناء. *A woman cleaning ingredients in the bowl.* 2. إمرأة تكسر ثلاثة بيضات في الإناء. *A woman breaks three eggs in a bowl.*	1.750
1. طفلة تعزف القيثارة. *A Child is playing harp.* 2. رجل يعزف القيثارة . *A man plays the harp.*	2.250
1. المرأة تقطع البصل الأخضر. *The woman chops green onions.* 2. إمرأة تقشر بصلة. *A woman peeling an onion.*	3.250
1. الأيل قفز فوق السياج. *The deer jumped over the fence.* 2. أيل يقفز فوق سياج الإعصار. *Deer Jumps Over Hurricane Fence*	4.800

Table 1.12: Example question pairs from the Arabic STS dataset. **Sentence Pair** column shows the two sentences. We also included their translations in the table. The translations were done by a native Arabic speaker. **Similarity** column indicates the annotated similarity of the two sentences.

1. **Bio-medical STS Dataset: BIOSSES** [12] - BIOSSES is the first and only benchmark dataset for biomedical sentence similarity estimation. [23]. The dataset comprises 100 sentence pairs, in which each sentence has been

[12]Bio-medical STS Dataset: BIOSSES can be downloaded from https://tabilab.cmpe.boun.edu.tr/BIOSSES/DataSet.html



(a) Word share against relatedness bins in Arabic STS train

(b) Word share against relatedness bins in Arabic STS test

Figure 1.15: Word share against relatedness bins in Arabic STS train and Spanish STS test. *Word Share* indicates the ratio between number of common words in the two sentences to total number of words in the two sentences against each *Relatedness* bins

Measure	Spanish STS Train		Spanish STS Test	
	Sent_1	**Sent_2**	**Sent_1**	**Sent_2**
Word Count Mean	31.23	31.02	9.03	9.34
Word Count STD	12.15	12.37	3.66	3.74
Word Count MAX	90	90	22	24
Word Count MIN	5	1	3	3

Table 1.13: Word count stats in Arabic STS training and Arabic STS testing.*STD* indicates the standard deviation and the other acronyms indicate the common meaning

selected from the TAC (Text Analysis Conference) Biomedical Summarisation Track- training dataset containing articles from the biomedical domain [13]. The sentence pairs have been evaluated by five different human experts that judged their similarity and gave scores ranging from 0 (no relation) to 4 (equivalent). The score range was described based on the guidelines of SemEval 2012 Task 6 on STS [9]. Besides the annotation instructions, example sentences from the bio-medical literature have been also provided to the annotators for each of the similarity degrees. To represent the similarity between two sentences we took the average of the scores provided by the five human experts. Table 1.14 shows few examples in the dataset. The machine learning models require to predict a value between 0-4 which reflects the similarity of the given bio medical sentence pair.

A dataset as small as this one can not be used by to train a supervised ML method, requiring alternative approaches such as unsupervised methods and transfer learning techniques which we will be exploring in the next few chapters.

1.3 Evaluation Metrics

While training a model is a key step, how the model generalises on unseen data is an equally important aspect that should be considered in every machine learn-

[13] Biomedical Summarisation Track is a shared task organised in TAC 2014 - https://tac.nist.gov/2014/BiomedSumm/

Figure 1.16: Relatedness distribution of BIOSSES. *Sentence Pairs* shows the number of sentence pairs that a certain *Relatedness bin* has.

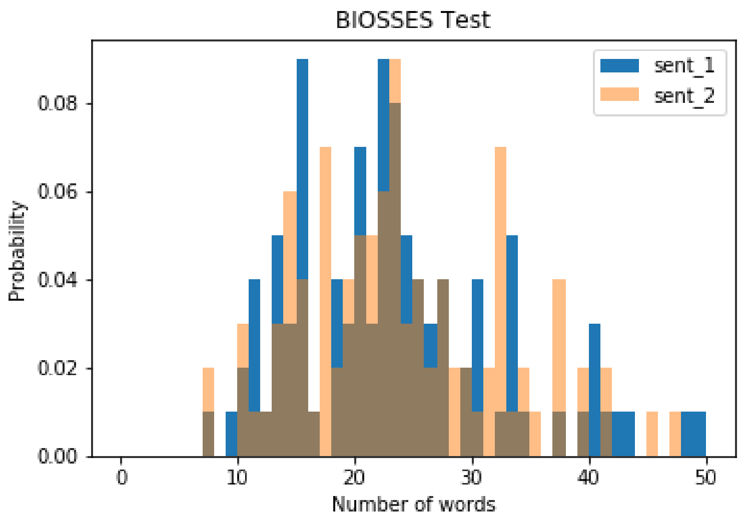


Figure 1.17: Normalised distribution of word count in BIOSSES. *Number of words* indicates the word count and *Probability* shows the total probability of a sentence with that word count appearing in the dataset.

Sentence Pair	**Similarity**
1. It has recently been shown that Craf is essential for Kras G12D-induced NSCLC. 2. It has recently become evident that Craf is essential for the onset of Kras-driven non-small cell lung cancer.	4
1. Up-regulation of miR-24 has been observed in a number of cancers, including OSCC. 2. In addition, miR-24 is one of the most abundant miRNAs in cervical cancer cells, and is reportedly up-regulated in solid stomach cancers.	3
1. These cells (herein termed TLM-HMECs) are immortal but do not proliferate in the absence of extracellular matrix (ECM) 2. HMECs expressing hTERT and SV40 LT (TLM-HMECs) were cultured in mammary epithelial growth medium (MEGM, Lonza)	1.4
1.The up-regulation of miR-146a was also detected in cervical cancer tissues. 2. Similarly to PLK1, Aurora-A activity is required for the enrichment or localisation of multiple centrosomal factors which have roles in maturation, including LATS2 and CDK5RAP2/Cnn.	0.2

Table 1.14: Example question pairs from the BIOSSES dataset. **Sentence Pair** column shows the two sentences. **Similarity** column indicates the averaged annotated similarity of the two sentences.

ing model. We need to know whether it actually works and, consequently, if we can trust its predictions. This is typically called as *evaluation.* All of the datasets that we introduced in the previous section has what we call a *test* set. The machine learning models need to provide their predictions for the test set and the predictions will be evaluated against the true values of the test set.

There are three common evaluation metrics that are employed in Semantic Textual Similarity tasks, which we explain in this section. We will be using them

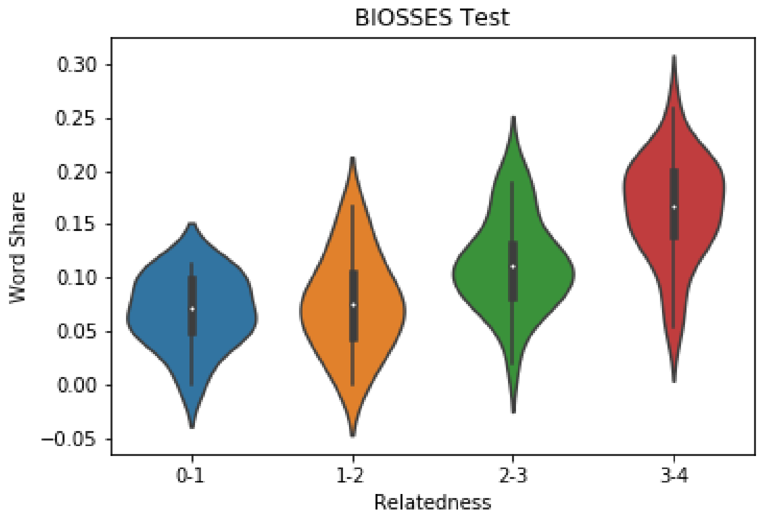


Figure 1.18: Word share against relatedness bins in BIOSSES. *Word Share* indicates the ratio between number of common words in the two sentences to total number of words in the two sentences against each *Relatedness* bins

to evaluate our models through out the first part of our research.

In the equations presented for each of the evaluation metrics, we represent the gold labels with X and predictions with Y. Therefore, a gold label in i^{th} position will be represented by X_i and a prediction in i^{th} position will be represented by Y_i.

1. **Pearson's Correlation Coefficient** - Correlation is a technique for investigating the relationship between two quantitative, continuous variables. Pearson's correlation coefficient (ρ) is a measure of the strength of the linear association between the two variables. A value of +1 is total positive linear correlation between the variables, 0 is no linear correlation, and -1 is total negative linear correlation.

   Pearson's Correlation Coefficient is one of the most common evaluation metrics in STS shared tasks [7, 9, 15, 16, 17, 18]. A machine learning model with a Pearson's Correlation Coefficient close to 1 indicates that the predictions of that model and gold labels have a strong positive linear correlation and therefore, it is a good model to predict STS. Pearson's Correlation Coefficient equation is shown in Equation 1.1 where *cov* is the covariance, σ_X is the standard deviation of X and σ_Y is the standard deviation of Y.

$$\rho = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y} \tag{1.1}$$

2. **Spearman's Correlation Coefficient** - Spearman's Correlation Coefficient (τ) is another common evaluation metric in STS shared tasks [7, 9, 15, 16, 17, 18]. It assesses how well the relationship between two variables can be described using a monotonic function. A monotonic relationship is a relationship that does one of the following:

   (a) as the value of one variable increases, so does the value of the other variable, *OR*,

   (b) as the value of one variable increases, the other variable value decreases.

But not exactly at a constant rate whereas in a linear relationship the rate of increase/decrease is constant. The fundamental difference between Pearson's Correlation Coefficient and Spearman's Correlation Coefficient is that the Pearson Correlation Coefficient only works with a linear relationship between the two variables whereas the Correlation Coefficient works with the monotonic relationships as well. Spearman's Correlation Coefficient equation is shown in Equation 1.2 where D_i is the pairwise distances of the ranks of the variables X_i and Y_i and n is the number of elements in X or Y.

$$\tau = 1 - \frac{6 \sum D_i^2}{n(n^2 - 1)} \tag{1.2}$$

3. **Root Mean Squared Error** - Both Pearson's Correlation Coefficient and Spearman's Correlation Coefficient works only when both gold labels(X) and predictions (Y) are continues. Therefore, in the datasets like Quora Question Pairs where the gold labels are discrete values, Root Mean Squared Error (RMSE) is preferred for evaluation than Correlation Coefficient values. RMSE measures the distance between the gold labels and the predictions. RMSE equation is shown in Equation 1.3 where n is the number of elements in X or Y.

$$rmse = \sqrt{(\frac{1}{n}) \sum_{i=1}^{n} (Y_i - X_i)^2} \tag{1.3}$$

1.4 Contributions

The main contributions of this part of the thesis are as follows.

1. In each chapter, we cover various techniques to compute semantic similarity at the sentence level that can benefit the applications in the machine

translation domain.

2. We propose a novel unsupervised STS method that outperforms current state of the arts unsupervised STS methods in all the English datasets, non-English datasets and datasets in other domains.

3. We propose a novel Siamese neural network architecture model which is efficient and outperforms current Siamese neural network architectures in all STS datasets.

4. We provide important resources to the community. The code of the each chapter as an open-source GitHub repository and the pre-trained STS models will be freely available to the community. The link to the GitHub repository and the models will be unveiled in the introduction section of the each chapter.

Chapter 2

Improving State of the Art Methods

The biggest challenge that the neural based architectures face when applied to STS tasks is the small size of datasets available to train them. As a result, in many cases the networks cannot be trained properly. Given the amount of human labour required to produce datasets for STS, it is not possible to have high quality large training datasets. As a result researches working in the field have also considered unsupervised methods for STS. Recent unsupervised approaches use pretrained word/sentence embeddings directly for the similarity task without training a neural network model on them. Such approaches have used cosine similarity on sent2vec [24], InferSent [25], Word Mover's Distance [26], Doc2Vec [27] and Smooth Inverse Frequency with GloVe vectors [28]. While these approaches have produced decent results in the final rankings of shared tasks, they have also provided strong baselines for the STS task.

This chapter explores the performance of three unsupervised STS methods - cosine similarity using average vectors, Word Mover's Distance [26] and cosine similarity using Smooth Inverse Frequency [28] and how to improve them using contextual word embeddings which will be explained more in Section 2.1.

We address four research questions in this chapter:

RQ1: Can contextual word embedding models like BERT be used to improve unsupervised STS methods?

RQ2: How well such an unsupervised method perform compared to other popular supervised/ unsupervised STS methods?

RQ3: Can the proposed unsupervised STS method be easily adopted in to different languages?

RQ4: How well the proposed unsupervised STS method perform in a different domain?

The main contributions of this chapter are as follows.

1. In the Related Work Section (Section 2.1), we cover three unsupervised STS techniques to compute semantic similarity at the sentence level.

2. We evaluate sentence en on three English STS datasets, two non-English STS datasets and a bio-medical STS dataset which were introduced in Chapter 1.

3. The code with the experiments conducted are publicly available to the community[1].

4. We published the findings in this chapter in Ranasinghe et al. [29].

The rest of this chapter is organised as follows. Section 2.1 describes the three unsupervised STS methods we experimented in this section. In section 2.2 we present the methodology, the contextual word embeddings we used followed

[1]The public GitHub repository is available on https://github.com/tharindudr/simple-sentence-similarity

by the results to the English datasets comparing with the baselines. Section 2.3 and Section 2.4 shows how our method can be applied to different languages and domains and their results. The chapter finishes with conclusions and ideas for future research directions in unsupervised STS methods.

2.1 Related Work

Given that a good STS metric is required for a variety of natural language processing fields, researchers have proposed a large number of such metrics. Before the shift of interest in neural networks, most of the proposed methods relied heavily on feature engineering. With the introduction of word embedding models, researchers focused more on neural representation for this task.

As we mentioned before, there are two main approaches which employ neural representation models: supervised and unsupervised. Unsupervised approaches use pretrained word/sentence embeddings directly for the similarity task without training a neural network model on them while supervised approaches uses a machine learning model trained to predict the similarity using word embeddings. Since this chapter focuses on unsupervised STS methods, this section would contain the previous research done on unsupervised STS methods.

The three unsupervised STS methods explored in this chapter: Cosine similarity on average vectors, Word Mover's Distance and Cosine similarity using Smooth Inverse Frequency are the most common unsupervised methods explored in STS tasks. Apart from them, cosine similarity of the output from Infersent [25], sent2vec [24] and doc2vec [27] have been used to represent the similarity between two sentences which we discuss in the next chapter.

2.1.1 Cosine Similarity on Average Vectors

The first unsupervised STS method that we considered to estimate the semantic similarity between a pair of sentences, takes the average of the word embeddings of all words in the two sentences, and calculates the cosine similarity between the resulting embeddings. This is a common way to acquire sentence embeddings from word embeddings. Obviously, this simple baseline leaves considerable room for variation. Researches have investigated the effects of ignoring stopwords and computing an average weighted by tf-idf in particular.

2.1.2 Word Mover's Distance

The second STS state of the arts method that we have considered is Word Mover's Distance introduced by Kusner et al. [26]. Word Mover's Distance uses the word embeddings of the words in two texts to measure the minimum distance that the words in one text need to "travel" in semantic space to reach the words in the other text as shown in Figure 2.1. Kusner et al. [26] shows that this is a good approach than vector averaging since this technique keeps the word vectors as it is through out the operation.

2.1.3 Cosine Similarity Using Smooth Inverse Frequency

The third and the last unsupervised STS method we have considered is to acquire sentence embeddings using Smooth Inverse Frequency proposed by Arora et al. [28] and then calculate the cosine similarity between those sentence embeddings.

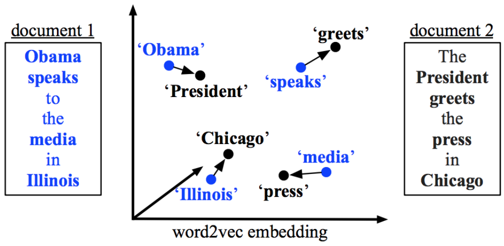


Figure 2.1: The Word Mover's Distance between two sentences

Semantically speaking, taking the average of the word embeddings in a sentence tends to give too much weight to words that are quite irrelevant. Smooth Inverse Frequency tries to solve this problem in two steps.

1. Weighting: Smooth Inverse Frequency takes the weighted average of the word embeddings in the sentence. Every word embedding is weighted by $\frac{a}{a+p(w)}$, where a is a parameter that is typically set to 0.001 and $p(w)$ is the estimated frequency of the word in a reference corpus.

2. Common component removal: After that, Smooth Inverse Frequency computes the principal component of the resulting embeddings for a set of sentences. It then subtracts their projections on first principal component from these sentence embeddings. This should remove variation related to frequency and syntax that is less relevant semantically.

As a result, Smooth Inverse Frequency downgrades unimportant words such as *but, just*, etc., and keeps the information that contributes most to the semantics of the sentence. After acquiring the sentence embeddings for a pair of sentences, the cosine similarity between those two vectors were taken to represent the similarity between them.

All of these STS methods are based on word embeddings/vectors. The main weakness of word vectors is that each word has the same unique vector regardless of the context it appears. Consider the word "play" which has several meanings, but in standard word embeddings such as GloVe [30], fastText [31] or Word2Vec [32] each instance of the word has the same representation regardless of the meaning which is used. For example the word 'bank' in two sentences - "I am walking by the river bank" and "I deposited money to the bank" would have the same embeddings which can be confusing for machine learning models. The recent introduction of contextualised word representations solved this problem by providing vectors for words considering their context too. In this way the word 'bank' in above sentences have two different embeddings. Contextual word embedding models have improved the results of many natural language processing tasks over traditional word embedding models [4, 33]. However, to the best of our knowledge they have not been applied on unsupervised STS methods.

Therefore, we explore how the contextualised word representations can improve the above mentioned unsupervised STS methods. We will explain the neural network architectures of these contextual word embeddings in Chapter 5. For this chapter, we considered these architectures as a black box where we just feed the words to get the embeddings. We considered these contextualised word representations mainly considering the popularity they had by the time we were doing the experiments.

1. **ELMo**[2] introduced by Peters et al. [33] use bidirectional language model (biLM) to learn both word (e.g., syntax and semantics) and linguistic context. After pre-training, an internal state of vectors can be transferred to downstream natural language processing tasks. ELMo vectors have been successfully used in many natural language processing tasks like text classification [34], named entity recognition [35] which motivated us to explore ELMo in unsupervised STS methods. Also, we were aware about the fact that ELMo has been pre-trained on different languages [36] and different domains [37] which will be easier when we are adopting our methodology for different languages and domains in Sections 2.3 and 2.4.

2. **BERT**[3] introduced by Devlin et al. [4] might probably be the most popular contextualised word embedding model. In contrast to ELMo which uses a shallow concatenation layer [4], BERT employs a deep concatenation layer. As a result BERT is considered a very powerful embedding architecture. BERT has been successfully applied in many natural language processing tasks like text classification [38], word similarity [39], named entity recognition [40], question and answering [41] etc. Similar to ELMo, BERT too has been widely adopted to different languages[4] such as Arabic [42], French [43], Spanish [44], Greek [45] etc. and different domains such as SciBERT [46], BioBERT [47], LEGAL-BERT [48] etc.

[2] More details about ELMo can be viewed on https://allennlp.org/elmo

[3] The GitHub repository of BERT is available on https://github.com/google-research/bert

[4] Information about pretrained BERT models for different languages can be found on https://bertlang.unibocconi.it/

3. **Flair**[5] is another type of popular contextualised word embeddings introduced in Akbik et al. [49]. It takes a different approach by using a character level language model rather than the word level language model used in ELMo and BERT. Flair also has been used successfully in natural language processing tasks such as named entity recognition [50], part-of-speech tagging [49] and has been widely adopted in to different languages and domains [49, 51].

Apart from using these contextual word embedding models individually we also considered **Stacked Embeddings** of these models together. Stacked Embeddings are obtained by concatenating different embeddings. According to Akbik et al. [49] stacking the embeddings can provide powerful embeddings to represent words. Therefore, we experimented with several combinations of Stacked Embeddings.

Even though these contextual word embedding models have shown promising results in many natural language processing tasks, to the best of our knowledge none of these contextual word representations has been applied on unsupervised STS methods.

2.2 Improving State of the Art STS Methods

As mentioned before we applied different contextual word embeddings on three unsupervised STS methods and their variants. First we experimented with English STS datasets we explained in Section 1.2. Our implementation was based

[5]The GitHub repository of Flair is available on `https://github.com/flairNLP/flair`

on *Flair-NLP* Framework [52] which makes it easier to switch between different word embedding models when acquiring word embeddings. Also *Flair-NLP* has their own model zoo of pre-trained models to allow researchers to use state-of-the-art NLP models in their applications. For English, all of these contextualised word embedding models come with different variants like *small, large etc.*. Usually the larger models provide a better accuracy since they have been trained on a bigger dataset compared to the smaller models. However, this comes with the disadvantage that these larger models are resource-intensive than the smaller models. In order to achieve a better accuracy, we used the largest model available in each contextual word embedding models. We will describe them in the following paragraphs.

For ELMo we used the 'original (5.5B)' pre-trained model provided in Peters et al. [33] which was trained on a dataset of 5.5B tokens consisting of Wikipedia (1.9B) and all of the monolingual news crawl data from WMT[6] 2008-2012 (3.6B). Peters et al. [33] mentions that ELMo original (5.5B) has slightly higher performance than other ELMo models and recommend it as a default model. Using this model we represented each word as a vector with a size of 3072 values.

For BERT we used the 'bert-large-cased' pre-trained model. Compared to the 'bert-base-cased' model, this model provided slightly better results in all the NLP tasks experimented in Devlin et al. [4]. We represented each word as a 4096 lengthened vector using this model.

[6]WMT: Workshop on Statistical Machine Translation is a leading conference in NLP that is being organised annually.

As suggested in Akbik et al. [49] the recommended way to use Flair embeddings is to stack pre-trained 'news-forward' flair embeddings and pre-trained flair 'news-backward' embeddings with GloVe [53] word embeddings. We used the stacked model to represent each word as a 4196 lengthened vector.

As mentioned before we also considered stacked embeddings of ELMo and BERT. For this we used pre-trained 'bert-large-uncased' model and 'original (5.5B)' pre-trained ELMo model to represent each word as a 4096 + 3072 vector.

In order to compare the results of contextualised word embeddings, we used a standard word representation model in each experiment as a baseline. In this research we used Word2vec embeddings [54] pre-trained on Google news corpus[7]. We represented each word as a 300 lengthened vector using this model.

In the following list we show the performance of each unsupervised STS method with contextual word embeddings on different English STS datasets.

1. **Cosine Similarity on Average Vectors** - The first unsupervised STS method we tried to improve using contextual word embeddings is Cosine Similarity on Average Vectors which we explained on Section 2.1. Table 2.1 shows the results for SICK dataset, Table 2.2 shows the results for STS 2017 dataset and Table 2.3 shows the results for Quora Question Pairs dataset. In order to compare our results with the other systems, we conducted the experiments only on the test data of the three mentioned datasets. Since this method leaves considerable room for variation, we have investigated

[7]Pretrained Word2vec can be downloaded from https://code.google.com/archive/p/word2vec/

the following variations and reported their results in each table.

(a) All the word vectors were considered for averaging. Results are shown in column I of Tables 2.1, 2.2 and 2.3

(b) All the word vectors except the vectors for stop words were considered for averaging. Column II of Tables 2.1, 2.2 and 2.3 shows the results.

(c) All the word vectors were weighted from its tf-idf scores and considered averaging. Results are shown in column III of Tables 2.1, 2.2 and 2.3

(d) Stop words were removed first and remaining word vectors were weighted from its tf-idf scores and considered averaging. Column IV of Tables 2.1, 2.2 and 2.3 shows the results.

	I		**II**		**III**		**IV**	
Model	ρ	τ	ρ	τ	ρ	τ	ρ	τ
Word2vec	**0.730**†	0.624	**0.714**	0.583	**0.693**	0.570	**0.687**	0.555
ELMo	0.669	0.592	0.693	0.603	0.676	**0.579**	0.668	**0.572**
Flair	0.646	0.568	0.670	0.562	0.644	0.535	0.643	0.531
BERT	0.683	0.633	0.686	0.606	0.557	0.552	0.539	0.538
ELMo $\bigoplus$ *BERT*	0.696	**0.634**†	0.702	**0.614**	0.607	0.562	0.591	0.551

Table 2.1: Results for SICK dataset with Vector Averaging. I, II, III and IV indicates the different variations as explained before. For each word embedding model, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported on all variations between the predicted values and the gold labels of the test set. $\bigoplus$ indicates a stacked word embedding model. Best result in each variation is marked in **Bold**. Best result from all the variations is marked with †.

From the results in Table 2.1, 2.2 and 2.3 there is no clear indication that

	I		II		III		IV	
Model	ρ	τ	ρ	τ	ρ	τ	ρ	τ
Word2vec	**0.625**†	0.583	0.609	**0.635**	**0.640**	**0.591**	**0.588**	**0.573**
ELMo	0.575	0.574	**0.618**	0.609	0.374	0.395	0.352	0.376
Flair	0.411	0.444	0.584	0.586	0.325	0.374	0.336	0.386
BERT	0.575	0.574	0.555	0.588	0.355	0.401	0.309	0.386
ELMo $\bigoplus$ *BERT*	0.600	**0.597**†	0.591	0.608	0.391	0.413	0.354	0.398

Table 2.2: Results for STS 2017 dataset with Vector Averaging. I, II, III and IV indicates the different variations as explained before. For each word embedding model, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported on all variations between the predicted values and the gold labels of the test set. $\bigoplus$ indicates a stacked word embedding model. Best result in each variation is marked in **Bold**. Best result from all the variations is marked with †.

	I	II	III	IV
Model	RMSE	RMSE	RMSE	RMSE
Word2vec	**0.621**	**0.591**†	**0.646**	**0.607**
ELMo	0.629	0.615	0.652	0.649
Flair	0.720	0.711	0.743	0.735
BERT	0.651	0.643	0.673	0.662
ELMo $\bigoplus$ *BERT*	0.625	0.611	0.650	0.647

Table 2.3: Results for QUORA dataset with Vector Averaging. I, II, III and IV indicates the different variations as explained before. For each word embedding model, Root Mean Squared Error (RMSE) is reported on all variations. $\bigoplus$ indicates a stacked word embedding model. Best result in each variation is marked in **Bold**. Best result from all the variations is marked with †.

contextualised word embeddings perform better than the standard word embeddings. In all the datasets considered, the best result was provided by Word2vec.

All the contextualised word embedding models we considered have more than 3000 dimensions for the word representation which is higher than the number of dimensions for the word representation we had for standard embeddings - 300. As the vector averaging model is highly dependent on the number of dimensions that a vector can have, the curse of dimensionality might be the reason for the poor performance of contextualised word embeddings in state of the art STS methods.

2. **Word Mover's Distance** - As the second unsupervised STS method we experimented with Word Mover's Distance explained on Section 2.1. Similar to the average vectors, we compared having contextualised word embeddings in the place of traditional word embeddings in Word Mover's Distance. Table 2.4 shows the results for SICK dataset. Table 2.5 shows the results for STS 2017 dataset and Table 2.6 shows the results for Quora Questions Pairs dataset. We have investigated the effects of considering/ ignoring stop words before calculating the word mover's distance which are detailed below.

   (a) Considering all the words to calculate the Word Mover's Distance. Results are shown in column I of Tables 2.4, 2.5 and 2.6

   (b) Removing stop words before calculating the Word Mover's Distance. Column II of Tables 2.4, 2.5 and 2.6 shows the results.

   As depicted in table contextualised word representations could not improve Word Mover's method too over standard word representations. Even though, ELMo $\bigoplus$ BERT model outperforms Word2vec in SICK and STS 2017 dataset with regard to Spearman Correlation (τ) there is no clear indication that contextual word representations would outperform standard

	I		II	
Model	ρ	τ	ρ	τ
Word2vec	**0.730**†	0.624	**0.714**	0.583
ELMo	0.669	0.592	0.693	0.603
Flair	0.646	0.568	0.670	0.562
BERT	0.683	0.633	0.686	0.606
ELMo $\bigoplus$ *BERT*	0.696	**0.634**†	0.702	**0.614**

Table 2.4: Results for SICK dataset with Word Mover's Distance. I and II indicates the different variations as explained before. For each word embedding model, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported on all variations between the predicted values and the gold labels of the test set. $\bigoplus$ indicates a stacked word embedding model. Best result in each variation is marked in **Bold**. Best result from all the variations is marked with †.

	I		II	
Model	ρ	τ	ρ	τ
Word2vec	**0.625**†	0.583	0.609	**0.635**
ELMo	0.575	0.574	**0.618**	0.609
Flair	0.411	0.444	0.584	0.586
BERT	0.575	0.574	0.555	0.588
ELMo $\bigoplus$ *BERT*	0.600	**0.597**†	0.591	0.608

Table 2.5: Results for STS 2017 dataset with Word Mover's Distance. I and II indicate the different variations as explained before. For each word embedding model, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported on all variations between the predicted values and the gold labels of the test set. $\bigoplus$ indicates a stacked word embedding model. Best result in each variation is marked in **Bold**. Best result from all the variations is marked with †.

word representations in Word Mover's method. Since the travelling distance is dependent on number of dimensions, the curse of dimensionality might be the reason for the poor performance of contextualised word representations in this scenario too.

3. **Smooth Inverse Frequency** As the third and the final unsupervised STS method we experimented with Smooth Inverse Frequency explained on

	I	II
Model	RMSE	RMSE
Word2vec	**0.621**	**0.591**†
ELMo	0.629	0.615
Flair	0.720	0.711
BERT	0.651	0.643
ELMo $\bigoplus$ *BERT*	0.625	0.611

Table 2.6: Results for QUORA dataset with Word Mover's Distance. I and II indicate the different variations as explained before. For each word embedding model, Root Mean Squared Error (RMSE) is reported on all variations. $\bigoplus$ indicates a stacked word embedding model. Best result in each variation is marked in **Bold**. Best result from all the variations is marked with †.

Section 2.1. Similar to the previous STS methods, we compared having contextualised word embeddings in the place of traditional word embeddings in Smooth Inverse Frequency method. Since the Smooth Inverse Frequency method takes care of stop words, we did not consider any variations that we experimented with previous STS methods. Table 2.7 shows the results for SICK dataset. Table 2.8 shows the results for STS 2017 dataset and Table 2.9 shows the results for Quora Questions Pairs dataset.

Model	ρ	τ
Word2vec	0.734	0.632
ELMo	0.740	0.654
Flair	0.731	0.634
BERT	0.746	0.661
ELMo $\bigoplus$ *BERT*	0.753†	0.669†

Table 2.7: Results for SICK dataset with Smooth Inverse Frequency. For each word embedding model, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported between the predicted values and the gold labels of the test set. $\bigoplus$ indicates a stacked word embedding model. Best result from all the variations is marked with †.

Model	ρ	τ
Word2vec	0.638	0.601
ELMo	0.641	0.609
Flair	0.639	0.606
BERT	0.650	0.612
ELMo $\bigoplus$ *BERT*	$0.654^{\dagger}$	$0.616^{\dagger}$

Table 2.8: Results for STS 2017 dataset with Smooth Inverse Frequency. For each word embedding model, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported between the predicted values and the gold labels of the test set. $\bigoplus$ indicates a stacked word embedding model. Best result from all the variations is marked with †.

Model	RMSE
Word2vec	0.599
ELMo	0.585
Flair	0.589
BERT	0.572
ELMo $\bigoplus$ *BERT*	$0.566^{\dagger}$

Table 2.9: Results for QUORA dataset with Smooth Inverse Frequency. For each word embedding model, Root Mean Squared Error (RMSE) is reported. $\bigoplus$ indicates a stacked word embedding model. Best result is marked with †.

As can be seen in the results, unlike the previous unsupervised STS methods, contextualised word embeddings improved the results in Smooth Inverse Frequency method compared to the standard word embeddings in all the three datasets considered. It can be observed that Smooth Inverse Frequency method is less sensitive to the number of dimensions in the word embedding model as it has a common component removal step and due to this reason, contextualised word embedding models does not suffer the *Curse of dimensionality* [55] with Smooth Inverse Frequency. In all of the datasets, the stacked embedding model of ELMo and BERT (ELMo $\bigoplus$

BERT) performed best. Further evaluating, from all the unsupervised STS methods we experimented including Vector Averaging and Word Movers Distance too, ELMo $\bigoplus$ BERT with the Smooth Inverse Frequency method provided the best results. With these observations, we address our *RQ1*, contextualised embeddings can be used to improve the unsupervised STS methods. Even though the contextual word embedding models did not improve the results in Vector Averaging and Word Movers Distance, there was clear improvement when they were applied on Smooth Inverse Frequency.

With regard to our *RQ2: How well the proposed unsupervised STS method performs when compared to various other STS methods?*, we compared our best results of the SICK dataset to the results in the SemEval 2014 Task 1 [7] which was the original task that the SICK dataset was initiated as we mentioned before. Our unsupervised method had 0.753 Pearson correlation score, whilst the best result in the competition had 0.828 Pearson correlation [7]. Our approach would be ranked on the ninth position from the top results out of 18 participants, and it is the best unsupervised STS method among the results [7]. Our method even outperformed systems that rely on additional feature generation (e.g. dependency parses) or data augmentation schemes. For example, our method is just above the UoW system which relied on 20 linguistics features fed in to a Support Vector Machine and obtained a 0.714 Pearson correlation [11]. Compared to these complex approaches our simple unsupervised approach provides a strong baseline to STS tasks. This answers our *RQ2*, that the proposed unsupervised STS method is competitive with the other supervised and unsupervised STS methods.

2.3 Portability to Other Languages

Our **RQ3** targets the multilinguality aspect of the proposed approach; *How well the proposed unsupervised STS method performs in different languages?*. To answer this, we evaluated our method in Arabic STS and Spanish STS datasets that were introduced in Chapter . Our approach has the advantage that it does not rely on language dependent features and it does not need a training set as the approach is unsupervised. As a result, the approach is easily portable to other languages given the availability of ELMo and BERT models in that particular language.

As the contextual word embedding models, for ELMo embeddings, we used the Arabic and Spanish Elmo models released by Che et al. [36]. Che et al. [36] have trained ELMo models for 44 languages including Arabic and Spanish using the same hyperparameter settings as Peters et al. [33] on Common Crawl and a Wikipedia dump of each language[8]. The models are hosted in NLPL Vectors Repository [56][9]. As for BERT we used the "BERT-Base, Multilingual Cased" model [4] which has been built on the top 100 languages with the largest Wikipedias that includes Arabic and Spanish languages too. Similar to the English experiments, we conducted the experiments through the *Flair-NLP* Frame-

[8]The GitHub repository for the ELMo for many languages project is available on https://github.com/HIT-SCIR/ELMoForManyLangs

[9]More information on the NLPL Vectors Repository is available on http://wiki.nlpl.eu/index.php/Vectors/home

work [52]. In order to compare the results, as traditional word embeddings, we used AraVec [57] [10] for Arabic and Spanish 3B words Word2Vec Embeddings [58][11] for Spanish.

Similar to the English datasets, from the unsupervised STS methods we considered, Smooth Inverse Frequency with ELMo and BERT stacked embeddings gave the best results for both Arabic and Spanish datasets. For Arabic our approach had 0.624 Pearson correlation whilst the best result [59] in the competition had 0.754 Pearson correlation [14]. Our approach would rank eighteenth out of 49 teams in the final results. Similar to English, our approach has the best result for an unsupervised method and surpasses other complex supervised models. For example Kohail et al. [60] proposes a supervised approach, which combines dependency graph similarity and coverage features with lexical similarity measures using regression methods and scored only 0.610 Pearson correlation. This shows that the proposed unsupervised STS method outperforms this supervised STS method.

For Spanish, our approach had 0.712 Pearson correlation whilst the best result [61] in the competition had 0.855 Pearson correlation [14]. Our approach would rank sixteenth out of 46 teams in the final results, which is the best result for an unsupervised approach. As with the English model, this one also surpasses other complex supervised models. For example Barrow and Peskov [62] uses a

[10] AraVec has been trained on Arabic Wikipedia articles. The models are available on `https://github.com/bakrianoo/aravec`

[11] Spanish 3B words Word2Vec Embeddings have been trained on Spanish news articles, Wikipedia articles and Spanish Boletín Oficial del Estado (BOE; English: Official State Gazette). The model is available on `https://github.com/aitoralmeida/spanish_word2vec`

supervised machine learning algorithm with word embeddings and scored only 0.516 Pearson correlation. Our fairly simple unsupervised approach outperform this supervised method by a large margin.

These findings answer our *RQ3*; the proposed unsupervised STS method can be successfully applied to other languages and it is very competitive even with the supervised methods.

2.4 Portability to Other Domains

In order to answer our *RQ4*; how well the proposed unsupervised STS method can be applied in different domains, we evaluated our method on Bio-medical STS dataset explained in 1. As we mentioned before Bio-medical STS dataset does not have a training set. Therefore, only the unsupervised approaches can be applied on this dataset which provides an ideal opportunity for the STS method we introduced in this Chapter.

For the experiments, as the contextual word embedding models, we used BioELMo [37][12], BioBERT [47][13] and BioFLAIR [51][14]. Additionally, to compare the performance with standard word embeddings, we used BioWordVec [63][15]. Same as English and multilingual experiments, Smooth Inverse Frequency with ELMo and BERT stacked embeddings performed best with this dataset too. It had

[12]BioELMo is the biomedical version of ELMo, pre-trained on PubMed abstracts. The model is available on https://github.com/Andy-jqa/bioelmo

[13]BioBERT has trained BERT on PubMed abstracts. The model is available on https://github.com/dmis-lab/biobert

[14]BioFLAIR is FLAIR embeddings trained on PubMed abstracts. The model is available on https://github.com/shreyashub/BioFLAIR

[15]BioWordVec has trained word2vec on a combination of PubMed and PMC texts. The model is availble on https://bio.nlplab.org/

0.708 Pearson correlation, whilst the best performing method had 0.836 Pearson correlation. This would rank our approach seventh out of 22 teams in the final results of the task [23].

It should be also noted that it outperforms many complex methods that sometimes uses external tools too. As an example, the UBSM-Path approach is based ontology based similarity which uses METAMAP [64] for extracting medical concepts from text and our simple unsupervised approach outperform them by a significant margin. UBSM-Path only has 0.651 Pearson correlation and compared to that our simple STS method based on contextual embeddings outperform them.

This answers our fourth and the final *RQ*; the proposed unsupervised STS method can be successfully applied in to other domains and it is very competitive with the available STS methods.

2.5 Conclusions

This chapter experimented three unsupervised STS methods namely cosine similarity using average vectors, Word Mover's Distance and cosine similarity using Smooth Inverse Frequency with contextualised word embeddings for calculating semantic similarity between pairs of texts and compared them with other unsupervised/ supervised approaches. Contextualised word embeddings could not improve cosine similarity using average vectors and Word Mover's Distance methods, but the results when using Smooth Inverse Frequency method were improved significantly with contextualised word embeddings, instead of standard word embeddings. Further more we learned that stacking ELMo and BERT provides a strong word representation rather than individual representations of ELMo and BERT. The results indicated that calculating cosine similarity using Smooth Inverse Frequency with stacked embeddings of ELMo and BERT is the best unsupervised method from the available approaches. Also, our approach finished on the top half of the final results list in the SICK dataset surpassing many complex and supervised approaches.

Our approach was also applied in the Arabic, Spanish and Bio-medical STS tasks, where our simple unsupervised method finished on the top half of the final result list in all the cases outperforming many supervised/ unsupervised STS methods. Therefore, given our results we can safely assume that regardless of the language or the domain cosine similarity using Smooth Inverse Frequency with stacked embeddings of ELMo and BERT will provide a simple but strong unsupervised method for STS tasks.

Contextual word embedding models are getting popular day by day due to their superior performance compared to standard word embedding models. Contextual word embedding models are available even in low resource languages like Assamese [65], Hebrew [66], Odia [65], Yoruba [67], Twi [67] etc. Very soon, contextual word embedding models would be available in all the languages where standard word embedding models available. Therefore, we can conclude that the unsupervised STS method we introduced in this chapter will be beneficial to many languages and domains.

As future work, the experiments can be extended in to other BERT like contextual word embedding models such as XLNet [5], RoBERTa [6], SpanBERT

[68] etc. One drawback of using Contextual word embedding models is that most of the pretrained models only support 512 maximum number of tokens which would be problematic when encoding longer sequences. Therefore, STS with long sequences can be explored with recently released contextual word embedding models like Longformer [69] and Big Bird [70] that supports encoding longer sequences than 512 maximum number of tokens. Taking advantage from the fact that this method is unsupervised and does not need a training dataset, it can further expanded in to many languages and domains as future work.

Chapter 3

Exploring STS with Sentence Encoders

3.1 Introduction

The main goal of a sentence encoder is to map a variable-length text to a fixed-length vector representation. In basic terms, a sentence encoder would take a sentence or text as the input and would output a vector. This vector encodes the meaning of the sentence and can be used for downstream tasks such as text classification, text similarity etc. In these down stream tasks, the sentence encoder is often considered as a black box where the users use the sentence encoder to get sentence embeddings without knowing what exactly happens in the encoder itself.

Ideally, the approaches we experimented in Chapter 2 like Vector Averaging, Smooth Inverse Frequency [28] can also be considered as sentence encoders since in those approaches the input is a variable-length text and the output is a fixed-length vector. However, these approaches have major drawbacks in representing sentences. One such drawback is these approaches do not care about the word order. If you consider two sentences *"Food is good but the service is bad"* and *"Food is bad but the service is good"*, would have the same embeddings from these approaches even though the meaning of these two sentences is completely different. Another drawback is those approaches lose information in the vector aggregation process. If you consider two sentences *"It is great"* and *"It is not great"*, Vector Averaging and Smooth Inverse Frequency would give similar sentence embeddings as there is only one word difference in the two sentences. Even though this affect can be minimised using techniques like TF/IDF weighting that we explored in Chapter 2, a different approach would be to train end-to-end models to get sentence embeddings. These models are commonly addressed as *sentence encoders* in NLP community.

Over the years, various sentence encodes like Sent2vec [24], Infersent [71], Universal Sentence Encoder [25] have been proposed. Even though most of these sentence encoders have sophisticated architectures, since many are using them only as a black box to get the sentence embeddings, they have been very popular in the NLP community. As the sentence encoders provide good quality sentence embeddings efficiently, the word embedding aggregation methods like Vector Averaging, Smooth Inverse Frequency have been often overlooked by the community in favour of sentence encoders.

Once you have the sentence embedding from a sentence encoder, using them to calculate STS is an easy task. Since these sentence embeddings are already semantically powerful, a simple vector comparison technique like cosine similarity between the embeddings can be used to calculate the STS of the two sentences. Therefore, pre-trained sentence encoders can be used as an unsupervised STS method. However, even though the sentence encoders have been commonly used in STS tasks, there is no comprehensive study done on them. Since most of the researchers are using sentence encoders as a black box in many applications, they don't understand the limitations of using them. In this chapter we are addressing this gap by exploring sentence encoders in different STS datasets adopting them in different languages and domains. With a study like this we can understand the limitations of the sentence encoders and when not to use them.

We address three research questions in this chapter:

RQ1: How well the sentence encoders perform in English STS datasets?

RQ2: Can the sentence encoders be easily adopted in different languages?

RQ3: How well the sentence encoders perform in different domains?

The main contributions of this chapter are as follows.

1. In the Related Work Section (Section 3.2), we discuss three sentence encoders that are popular in the NLP community.

2. We evaluate these three sentence encoders on three English STS datasets, two non-English STS datasets and a bio-medical STS dataset which were introduced in Chapter 1.

3. The code with the experiments conducted are publicly available to the community[1].

The rest of this chapter is organised as follows. Section 3.2 describes the three sentence encoders we experimented in this section. In Section 3.3 we present the experiments we conducted with the three sentence encoders in English STS

[1]The public GitHub repository is available on https://github.com/tharindudr/simple-sentence-similarity

datasets followed by the results comparing with the other unsupervised STS methods. Section 3.4 and Section 3.5 shows how sentence encoders can be applied to different languages and domains and their results. The chapter finishes with conclusions and ideas for future research directions in sentence encoders.

3.2 Related Work

As we mentioned before, the sentence encoders are popular with the NLP community. The three sentence encoders explored in this chapter: Sent2vec [24], Infersent [25] and Universal Sentence Encoder [71] are the most common sentence encoders. Other than them there is also Doc2vec [72] which is can be considered as a sentence encoder. However, due to the various upgrades in different python libraries, the official pre-trained Doc2vec models are not working any more. Therefore, we only used following sentence encoders in our experiments.

Sent2vec Sent2vec presents a simple but efficient unsupervised objective to train distributed representations of sentences [24]. It can be thought as an extension of Word2vec (CBOW) to sentences. The key differences between CBOW and Sent2Vec are the removal of the input subsampling, considering the entire sentence as context, as well as the addition of word n-grams. With Sent2vec, sentence embedding is defined as the average of the word embeddings of its constituent words. The objective of Sent2vec is similar to CBOW; predict the missing word given the context [24].

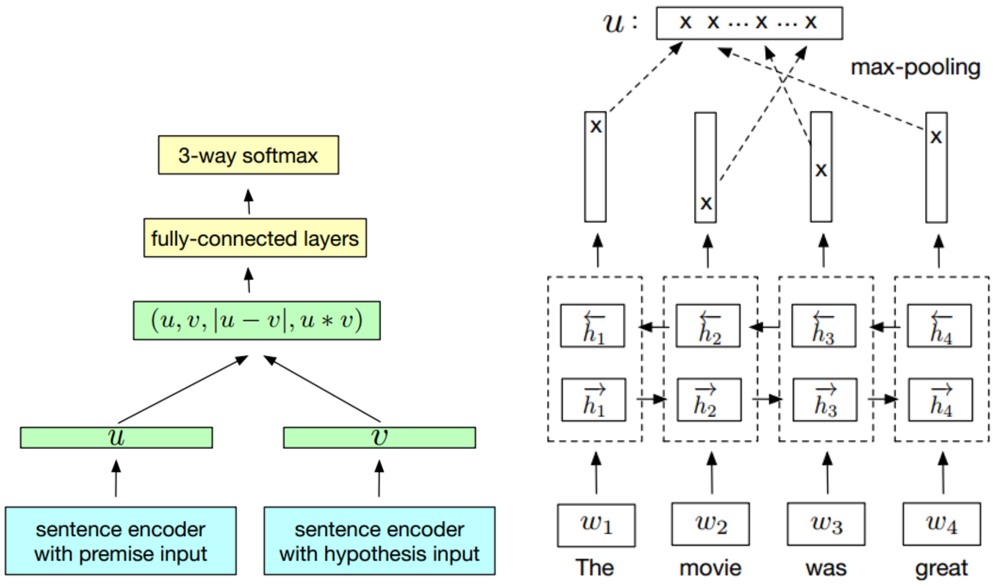
Sent2vec has officially released several pre-trained models to derive the sentence embeddings[2]. Also due to its unsupervised approach and simple objective function, Sent2vec has been adopted in different languages and domains too.

Infersent InferSent is an NLP technique for universal sentence representation developed by Facebook which uses supervised training to produce high quality sentence vectors [25]. The authors explore 7 different architectures for sentence encoding including LSTM [73], GRU [74], Bi directional LSTM [75] with mean/max pooling, Self-attentive network and Hierarchical Deep Convolutional Neural Network [25]. All of these models were trained for the natural language inference (textual entailment) task using the architecture in Figure 3.1a. They evaluate the quality of the sentence representation by using them as features in 12 different transfer tasks like Binary and multi-class text classification, semantic textual similarity, paraphrase detection etc. The results indicate that the BiLSTM with the max-pooling operation performs best on these tasks [25]. The architecture of this BiLSTM with the max-pooling model is shown in Figure 3.1b.

Facebook released two models to derive the sentence embeddings. One model is trained with GloVe [53] which in turn has been trained on text preprocessed with the PTB tokeniser and the other model is trained with fastText [31] which has been trained on text preprocessed with the MOSES tokeniser. We used both models in our experiments[3].

[2]The code and the pre-trained models are available on `https://github.com/epfml/sent2vec`

[3]The code and the pre-trained models are available on `https://github.com/facebookresearch/InferSent`



(a) General NLI Training Scheme. (b) Bi directional LSTM with max pooling

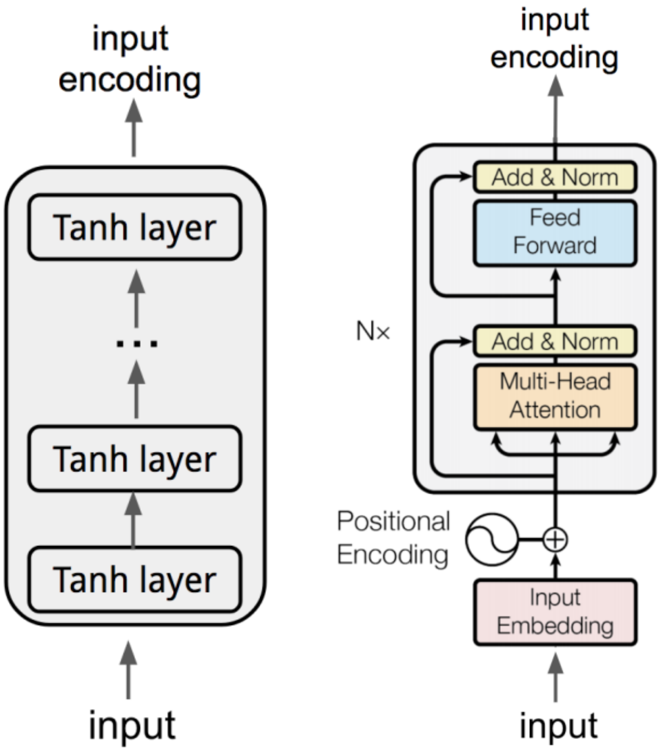
Figure 3.1: General NLI training scheme in Infersent with the best architecture; Bi directional LSTM with max pooling [25].

Universal Sentence Encoder The Universal Sentence Encoder [71] released by Google is the last sentence encoder we employed in this chapter. This is again an unsupervised sentence encoder. It comes with two versions i.e. one trained with Transformer encoder and other trained with Deep Averaging Network (DAN). Both architectures are outlined briefly below. The two have a trade-off of accuracy and computational resource requirement. While the one with Transformer encoder has higher accuracy, it is computationally more expensive. The one with DAN encoding is computationally less expensive but with slightly lower accuracy.

The original Transformer encoder model constitutes an encoder and decoder. Since our research is only focussed on encoding sentences to vectors, we only use its encoder part. The encoder is composed of a stack of N = 6 identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network. Cer et al. [71] also employed a residual connection around each of the two sub-layers, followed by layer normalisation. Since the model contains no recurrence and no convolution, for the model to make use of the order of the sequence, it must inject some information about the relative or absolute position of the tokens in the sequence, that is what the "positional encodings" does. The transformer-based encoder achieves the best overall transfer task performance. However, this comes at the cost of computing time and memory usage scaling dramatically with sentence length.

Deep Averaging Network (DAN) is much simpler where input embeddings for words and bi-grams are first averaged together and then passed through a feedforward deep neural network to produce sentence embeddings. The primary advantage of the DAN encoder is that computation time is linear in the length of the input sequence. With this sentence encoder too, we used both architectures in our experiments[4] Unlike the other sentence encoders, Google officially released two multilingual models for Universal Sentence Encoder.

[4]Pre-trained sentence encoder for transformer model is available on `https://tfhub.dev/google/universal-sentence-encoder-large` and pre-trained sentence encoder for DAN model is available on `https://tfhub.dev/google/universal-sentence-encoder`.



(a) DAN Architecture (b) Transformer Architecture

Figure 3.2: Two architectures in Universal Sentence Encoders.

3.3 Exploring Sentence Encoders in English STS

Adopting sentence encoders for STS is an easy task. If two embeddings from the two sentences are closer, the sentences are said to be semantically similar. As the approach, first the two sentences are passed to the sentence encoders to get the embeddings and the we calculate the cosine similarity between the resulting embeddings which represents the textual similarity of the two input sentences. To be clear, if the two vectors for two sentences X and Y are a and b correspondingly, we calculate the cosine similarity between a and b as of equation 3.1 and use that value to represent the similarity between the two sentences.

$$\begin{aligned} \cos(\mathbf{a}, \mathbf{b}) &= \frac{\mathbf{a}\mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|} \\ &= \frac{\sum_{i=1}^{n} \mathbf{a}_i \mathbf{b}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{a}_i)^2} \sqrt{\sum_{i=1}^{n} (\mathbf{b}_i)^2}} \end{aligned} \tag{3.1}$$

First we experimented with English STS datasets we explained in Section 1.2. For the experiments we used all the sentence encoders mentioned in Section 3.2. For **Sent2vec**, we used the pre-trained Sent2vec model, *sent2vec_wiki_bigrams* trained on English Wikipedia articles. Using that we could represent a sentence from a 700 dimensional vector. For **Infersent**, as we mentioned before, there are two pre-trained models available; *infersent1* which was trained using GloVe [53] and *infersent2* which was trained using fastText [31]. Both models have been trained on the SNLI dataset which consists of 570k human generated English sentence pairs, manually labelled with one of the three categories: entailment, contradiction and neutral [19]. Using that we could represent a sentence from a 512 dimensional vector. For **Universal Sentence Encoder**, we used universal-sentence-encoder (DAN architecture) and universal-sentence-encoder-large (Transformer architecture) which were trained on text resources like Wikipedia and news articles. With that too we could represent a sentence from a 512 dimensional vector.

We evaluated these three sentence encoders in three English STS datasets that we explained in Section 1.2; SICK, STS2017 and QUORA. Table 3.1 shows the results for SICK dataset, Table 3.2 shows the results for STS 2017 dataset and Table 3.3 shows the results for Quora Questions Pairs dataset.

Model	ρ	τ
ELMo $\bigoplus$ *BERT*	0.753	0.669
Sent2vec	0.759	0.672
Infersent1	0.763	0.679
Infersent2	0.769	0.684
USE (DAN)	0.772	0.695
USE (Transformer)	0.780†	0.721†

Table 3.1: Results for SICK dataset with sentence encoders. For each sentence encoder, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported between the predicted values and the gold labels of the test set. USE denotes Universal Sentence Encoder. Additionally, we report the results of the best model from Chapter 2; *ELMo* $\bigoplus$ *BERT*. Best result from all the methods is marked with †.

As can be seen in the results, Universal Sentence Encoder outperformed all other sentence encoders in all the English STS datasets. From the two architectures available in the Universal Sentence Encoder, Transformer architecture outperforms the DAN architecture as they have explained in their paper. Furthermore, it should be noted that in all three datasets, sentence encoders outper-

Model	ρ	τ
ELMo $\bigoplus$ BERT	0.654	0.616
Sent2vec	0.673	0.645
Infersent1	0.703	0.696
Infersent2	0.711	0.701
USE(DAN)	0.725	0.703
USE(Transformer)	0.744†	0.721†

Table 3.2: Results for STS 2017 dataset with sentence encoders. For each sentence encoder, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported between the predicted values and the gold labels of the test set. USE denotes Universal Sentence Encoder. Additionally, we report the results of the best model from Chapter 2; *ELMo $\bigoplus$ BERT*. Best result from all the methods is marked with †.

Model	RMSE
ELMo $\bigoplus$ BERT	0.566
Sent2vec	0.632
Infersent1	0.642
Infersent2	0.653
USE(DAN)	0.666
USE(Transformer)	0.686†

Table 3.3: Results for QUORA dataset with sentence encoders. For each sentence encoder model, Root Mean Squared Error (RMSE) is reported. USE denotes Universal Sentence Encoder. Additionally, we report the results of the best model from Chapter 2; *ELMo $\bigoplus$ BERT*. Best result is marked with †.

form the embedding aggregation based smooth inverse frequency method that performed best in Chapter 2. This concludes that sentence encoders generally perform better than embedding aggregation techniques in STS.

With these results, we can answer our **RQ1**, sentence encoders can be easily adopted and perform well in English STS tasks. However, most of these models are complex in nature which would result in more processing time/resources which can be chaotic in some situations.

3.4 Portability to Other Languages

Our **RQ2** targets the multilinguality aspect of the sentence encoders; *How well the sentence encoders perform in different languages?*. To answer this, we evaluated our method in Arabic STS and Spanish STS datasets that were introduced in Chapter 1. With these experiments, we identified a main weakness in sentence encoders; sentence encoders pre-trained on different languages are not easy to find.

If you consider **Infersent**, it was pre-trained using the SNLI dataset which consists of 570k human generated English sentence pairs, manually labelled with one of three categories: entailment, contradiction and neutral [19]. If someone is adopting **Infersent** for a different language other than English, they need to have a corpus similar to SNLI with a similar size. Annotating such a corpus for a different language would be challenging. Even though there are some attempts like XNLI [76] to create such a corpus, the number of annotated instances are very limited. This makes it difficult to adopt **Infersent** in other languages which is a clear limitation of the Infersent architecture.

However, the other two sentence encoders we experimented in this Chapter; Sent2vec and Universal Sentence Encoder are in a better position in multilingualism compared to Infersent as they don't require a large annotated corpus like SNLI. Both of those sentence encoders have been trained on unsupervised textual data which will be easy to find in most of the languages. However, still they need powerful computing resources to train the models which is a challenge when adopting these sentence encoders to different languages.

For **Sent2Vec**, there was no Arabic pre-trained model available. However, there is a Spanish Sent2vec model[5] available which is pre-trained on Spanish Unannotated Corpora. Using that we could represent a Spanish sentence with a 700 dimensional vector. For **Universal Sentence Encoder**, there is a multilingual version which supports 16 languages[6] including Arabic and Spanish [77]. This multilingual model is available in both architecture in Universal Sentence Encoder; DAN and Transformer[7]. Using that we could represent the Arabic and Spanish sentences with a 512 dimensional vector. As we mentioned before, for **Infersent**, we could not find any pre-trained models that supports either Arabic nor Spanish. Therefore, we did not use Infersent in our multilingual experiments. The Arabic and Spanish STS results with the mentioned sentence encoders are available in Table 3.4.

As can be seen in results, similar to the English datasets, from the sentence encoders we considered, Universal Sentence Encoder with the Transformer architecture gave the best results for both Arabic and Spanish datasets. From the two architectures available in the Universal Sentence Encoder, Transformer architecture outperforms the DAN architecture in both languages. Furthermore,

[5]The pre-trained model is available on https://github.com/BotCenter/spanish-sent2vec

[6]The model currently supports Arabic, Chinese-simplified, Chinese-traditional, English, French, German, Italian, Japanese, Korean, Dutch, Polish, Portuguese, Spanish, Thai, Turkish and Russian.

[7]The multilingual Universal Sentence Encoder with DAN architecture is available on https://tfhub.dev/google/universal-sentence-encoder-multilingual/3 and the multilingual Universal Sentence Encoder with Transformer architecture is available on https://tfhub.dev/google/universal-sentence-encoder-multilingual-large/3.

Language	Sentence Encoder	ρ	τ
Arabic	*ELMo $\bigoplus$ BERT*	0.624	0.589
	USE (DAN)	0.654	0.612
	USE (Transformer)	$0.668^{\dagger}$	$0.635^{\dagger}$
Spanish	*ELMo $\bigoplus$ BERT*	0.712	0.663
	USE (DAN)	0.723	0.6.682
	Sent2vec	0.725	0.688
	USE (Transformer)	$0.741^{\dagger}$	$0.702^{\dagger}$

Table 3.4: Results for Arabic and Spanish STS with different sentence encoders. For each sentence encoder, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported between the predicted values and the gold labels of the test set. USE denotes Universal Sentence Encoder. Additionally, we report the results of the best model from Chapter 2; *ELMo $\bigoplus$ BERT*. Best result for each language is marked with †.

it should be noted that in both languages, sentence encoders outperform the word embedding based smooth inverse frequency method that performed best in Chapter 2.

With these experiments, we can answer our **RQ2: How well the sentence encoders can be adopted in different languages?**. Adopting sentence encoders in different languages is challenging since there are no available pre-trained sentence encoder models for many languages. However, in the cases where they are available, it is very easy to use them in STS tasks and they provide good results compared to other unsupervised STS methods.

3.5 Portability to Other Domains

In order to answer our *RQ3*; how well the sentence encoders can be applied in different domains, we evaluated sentence encoders that we explained in this Chapter on the Bio-medical STS dataset explained in 1. As we mentioned before, Bio-medical STS dataset does not have a training set. Therefore, only the unsupervised approaches can be applied on this dataset which provides an ideal opportunity for the sentence encoders we experimented in this chapter.

However, we faced the same issue we faced in Arabic and Spanish STS experiments in Bio-medical STS experiments too. There are not many options when it comes to sentence encoders that were trained on Bio-medical domain [78]. For **Sent2vec**, we could find a pre-trained model in Bio-medical domain which was trained using PubMed data [79][8]. However, for other two sentence encoders, we could not find any available pre-trained sentence encoder models on Bio-medical domain. Therefore, for Universal Sentence Encoder and Infersent, we had to use pre-trained sentence encoder models trained on English texts for this experiments. The results are shown in Table 3.5.

Model	ρ
Infersent2	0.294
Infersent1	0.301
USE(DAN)	o.321
USE(Transformer)	0.345
ELMo $\bigoplus$ *BERT*	0.708
BioSentVec [79]	0.810†

Table 3.5: Results for BIOSSES dataset with different sentence encoders compared with top results reported for BIOSSES. Additionally, we report the results of the best model from Chapter 2; *ELMo* $\bigoplus$ *BERT*. For each variant, Pearson Correlation (ρ) is reported between the predicted values and the gold labels of the test set. Best result is marked with †.

As can be observed in the results, sentence encoder trained on the Bio-medical

[8]The code and the pre-trained model is available on `https://github.com/ncbi-nlp/BioSentVec`

domain; BioSentVec [79] outperformed other approaches. In fact, when compared to the best results in the BIOSSES dataset BioSentVec outperforms all of them which means that BioSentVec provides the best result for BIOSSES. it should be noted that out of domain sentence encoders like Universal Sentence Encoder and Infersent, that were not trained on the Bio-medical domain performed very poorly in these experiments. The simple *ELMo* $\bigoplus$ *BERT* embedding aggregation based approach we experimented in Chapter 2 outperforms Universal Sentence Encoder and Infersent by a large margin. This can be due to the fact that, there is a large number of out-of-vocabulary words for these general sentence coders in the Bio-medical domain unlike the *ELMo* $\bigoplus$ *BERT* where we used the ELMo and BERT models trained on Bio-medical domain. We can conclude that sentence encoders can be successfully adopted for STS in different domains, however they won't succeed unless they are pre-trained on that particular domain.

With these finding we answer our **RQ3: Is it possible to adopt sentence encoders in different domains?**. We showed that it is possible to adopt sentence encoders in a different domain. However, it is difficult since pre-trained sentence encoder models are not common in most domains and the sentence encoders pre-trained on a general domain perform poorly on a specific domain like Bio-medical.

3.6 Conclusions

In this chapter we experimented another unsupervised STS method; sentence encoders. We explored three popular sentence encoders; Sent2vec, Infersent and Universal Sentence Encoder in three English STS datasets. The results show that sentence encoders outperforms other unsupervised STS methods. From the sentence encoders, Universal Sentence Encoder with the Transformer architecture performs best in all three datasets. Furthermore, we evaluated sentence encoders in different languages and domains. We experienced the biggest hurdle in adopting sentence encoders in different languages and domains; the pre-trained sentence encoders that support different languages and domains are not common compared to the word embedding/ contextual word embedding models available on those languages and domains. This is because the sentence encoders take a lot of time to train and some of the sentence encoders like Infersent require specific training data, which is hard to compile in most of the languages and domains. Also, since the applications of sentence encoders are limited compared to word embeddings/ contextual word embeddings people have not spend time on creating language specific and domain specific sentence encoders. However, when available they perform very well in the relevant tasks. Also we tried to use sentence encoders that were trained on a general domain in a different but specific domain like Bio-medical. This provided very poor results. Therefore, we can conclude that using sentence encoders on a different domain to their pre-trained domain won't provide good results.

Being an unsupervised STS method, the sentence encoders have the obvious advantage of not needing a training set for STS. However, they still need a pre-trained sentence encoding model, which is not available in most of the languages and domains. This is a major drawback in sentence encoders compared to the embedding aggregation methods since the word embedding/ contextual embedding models are commonly available in many languages and domains. This is worsened by the fact that sentence encoders perform very poorly on the domains that they did not see in the training process. This suggests that we should only use sentence encoders when available in a certain domain/ language, otherwise using embedding aggregation methods like Smooth Inverse Frequency would provide better results.

As future work, the experiments can be extended in to recently released sentence encoders like LASER [80]. LASER supports 93 languages including low resource languages like Kabyle, Malagasy, Sinhala etc. This should solve some of the multilingual issues we experienced with sentence encoders in this chapter. Very recently sentence encoders have been explored with Siamese neural architectures and Transformers. We discuss them in Chapters 4 and 5.

Chapter 4

Siamese Neural Networks for STS

4.1 Introduction

A *Siamese Neural Network* is a class of neural network architectures that contain two or more identical subnetworks which is usually employed in applications that requires comparisons between two or more inputs (signature verification, image similarity etc.). The network can take two or more inputs at the same time and they will be processed by different subnetworks at the same time. The subnetworks have the same configuration with the same parameters and weights. The training process is mirrored across all the subnetworks which means that the parameters remain same with all the subnetworks. Each subnetwork contains a traditional perceptron model like LSTM, CNN etc. The neural network compares the output of the two subnetworks through a distance metric like a cosine distance and the error is back-propagated. In the testing phase, the neural network predicts whether the two inputs are different through this similarity measure.

Siamese neural networks have been employed in many applications in different areas like signal processing [81, 82, 83, 84, 85, 86, 87, 88], biology [89, 90], chemistry and pharmacology [91], geometry [92], computer vision [93, 94, 95, 96, 97, 98, 99, 100, 101, 102], physics [103, 104], robotics [105, 106, 107], video processing [108, 109, 110, 111] etc. They have also been used in NLP tasks [112, 113, 114] including STS [115, 116, 117]. In fact, the best result in the SICK dataset [7] was provided by a Siamese neural network [117] which shows that state-of-the-art in STS is Siamese neural networks.

In addition to providing state-of-the-art results in STS, there are additional advantage in using Siamese neural networks. As we mentioned before, in Siamese neural networks, as the weights are shared across subnetworks there are fewer parameters to train, which in turn means they require less training data and less tendency to over-fit. Given the amount of human labour required to produce datasets for STS, Siamese neural networks can provide the ideal solution for the STS task. As the second advantage, the Siamese neural network architecture when trained on a STS task, can be adopted as sentence encoders. The output vector of the subnetwork is a semantically rich vector representation of the input sentence [117]. These advantages motivates us to explore Siamese neural network architectures more in STS.

We address four research questions in this chapter:

RQ1: Can existing state-of-the-art Siamese neural network architecture be modified to provide better STS results?

RQ2: Can the method further improved with transfer learning and data augmentation techniques?

RQ3: Can the proposed Siamese neural network be easily adopted in to different languages?

RQ4: How well the proposed Siamese neural network perform in a different domain?

The main contributions of this chapter are as follows.

1. We propose a GRU (Gated recurrent unit) based Siamese neural network that outperforms state-of-the-art LSTM based Siamese neural network in small STS datasets.

2. We propose a LSTM (Long short-term memory) and Self-attention based Siamese neural network that outperforms state-of-the-art LSTM based Siamese neural network in large STS datasets.

3. We propose further enhancements to the architecture using transfer learning and data augmentation.

4. We evaluate how well the proposed Siamese neural network architecture performs in different languages and domains.

5. The initial findings of this chapter is published in Ranasinghe et al. [118].

6. The code and the pre-trained models are publicly available to the community[1]

The rest of this chapter is organised as follows. Section 4.2 describes the past research done with Siamese neural networks. Section 4.3 discusses the methodology and the experiments done with three English STS datasets. Section 4.3.1 and 4.3.2 provides more experiments to improve the results. Experiments done with

[1] The public GitHub repository is available on https://github.com/tharindudr/Siamese-recurrent-rrchitectures

other languages and domains are shown in Section 4.4 and Section 4.5. The chapter finishes with conclusions and ideas for future research directions in Siamese neural networks.

4.2 Related Work

The Siamese neural networks have been very popular in the machine learning community. The first appearance of Siamese neural networks date back to 1994. It was first introduced by Bromley et al. [119] to detect forged signatures. By comparing two handwritten signatures, this Siamese neural network was able to predict if the two signatures were both original or if one was a forgery. Even before that, Baldi and Chauvin [93] introduced a similar artificial neural network able to recognize fingerprints, though by a different name.

Siamese neural networks have been applied in various applications after that. In audio and speech signal processing field, Thiolliere et al. [81] merged a dynamic-time warping based spoken term discovery (STD) system with a Siamese deep neural network for automatic discovery of linguistic units from raw speech, Manocha et al. [82] used a Siamese model to detect all the semantically similar audio clips in an input audio recording and Shon et al. [83] employed a Siamese model to recognize Arabic dialects from Arabic speech content found in media broadcasts. In biology, Zheng et al. [89] implemented a Siamese neural network to compare DNA sequences and recently Szubert et al. [90] present a Siamese neural network-based technique for visualization and interpretation of single-cell datasets. Image analysis is the field with the highest number of applications for the Siamese neural networks. Recognising fingerprints [93], similar image detection [94, 95, 96, 97, 98], face verification [99], gesture recognition [100], hand writing analysis [101] and patch matching [102] are some of them. As you can observe, all of these tasks involve in comparing two or more things.

Recently, Siamese neural networks have been employed in NLP too. Yih et al. [112] proposed similarity Learning via Siamese Neural Network (S2Net), a technique able to discriminatingly learn concept vector representations of text words. Kumar et al. [113] used Siamese neural network to recognize clickbaits in online media outlets. González et al. [114] proposed a natural language processing application of the Siamese neural network for extractive summarization, which means that their technique can extrapolate most relevant sentences in a document. Not limited to those applications Siamese neural networks have been implemented in STS tasks too in NLP. Das et al. [115] used a CNN based Siamese neural network to detect similar questions on question and answer websites such as Yahoo Answers, Baidu, Zhidao, Quora, and Stack Overflow. Neculoiu et al. [116] employed a Siamese neural network based on Bidirectional LSTMs to identify similar job titles. The baseline we used for this chapter; MALSTM [117] uses a LSTM based Siamese neural network to perform semantic textual similarity and it provides the best results for SICK dataset outperforming other STS methods like Tree-LSTMs [2]. They use the exponent of the negative Manhattan distance between two outputs from the two subnetworks as the similarity function. Due to the performance this can be considered as the state-of-the-art Siamese neural network for STS. However, this architecture leaves considerable room for variation which we exploit in this chapter as we explain in Section 4.3.

4.3 Exploring Siamese Neural Networks for STS

The basic structure of the Siamese neural network architecture used in our experiments is shown in Figure 4.1. It consists of an embedding layer which represents each sentence as a sequence of word vectors. This sequence of word vectors is then fed into a Recurrent Neural Network (RNN) cell which learns a mapping from the space of variable length sequences of 300-dimensional vectors into a 50 dimensional vector. The sole error signal back propagated during training, stems from the similarity between these 50 dimensional vectors, which can be also used as a sentence representation. Initially, the similarity function we used was based on Manhattan distance. To make sure that the prediction is between 0 and 1, we took the exponent of the negative Manhattan distance between 2 sentence representations. The similarity function was adopted from Mueller and Thyagarajan [117]. The proposed variants of our architecture are:

1. LSTM - Block A in Figure 4.1 contains a single LSTM cell. This is the architecture suggested by Mueller and Thyagarajan [117]

2. Bi-directional LSTM - Block A in Figure 4.1 contains a single Bi-directional LSTM cell. Bi-directional LSTM tends to understand the context better than Unidirectional LSTM [75].

3. GRU - Block A in Figure 4.1 contains a single GRU cell. GRUs have been shown to exhibit better performance on smaller datasets [74].

4. Bi-directional GRU - Block A in Figure 4.1 contains a single Bi-directional GRU cell. Bi-directional GRUs tend to understand the context better than Unidirectional GRU [120].

5. LSTM + Attention - Block A in Figure 4.1 contains a single LSTM cell with self attention [121].

6. GRU + Attention - Block A in Figure 4.1 contains a single GRU cell with self attention [121].

7. GRU + Capsule + Flatten - Block A in Figure 4.1 contains a GRU followed by a capsule layer and a flatten layer. Dynamic routing used between capsules performs better than a traditional max-pooling layer [122].

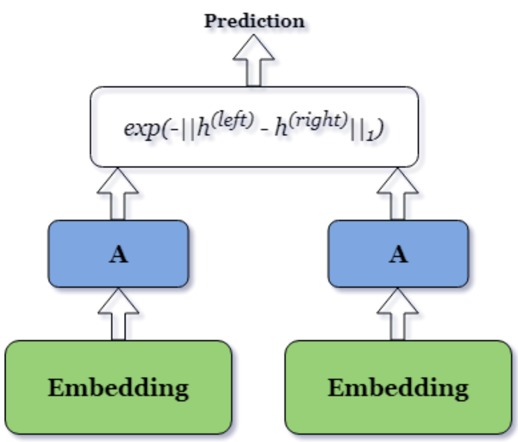


Figure 4.1: Basic structure of the Siamese neural network. Unit A is changed over the architectures.

As the word embedding model we used Word2vec embeddings [54] pre-trained on Google news corpus[2]. We represented each word as a 300 lengthened vector using this model. For the words that do not appear in this model we used a random vector. We evaluated all the above variations in the three English STS datasets we introduced in 1; SICK, STS 2017 and QUORA. We trained the Siamese models on the training sets on those datasets and evaluated them on the testing sets. The results are shown in Table 4.1, Table 4.2 and Table 4.3 respectively.

Model	ρ	τ
LSTM	0.802	0.733
Bi-LSTM	0.784	0.708
GRU	0.838†	0.780†
Bi-GRU	0.832	0.773
LSTM + Attention	0.827	0.765
GRU + Attention	0.818	0.751
GRU + Capsule + Flatten	0.806	0.733

Table 4.1: Results for SICK dataset with different variants of Siamese Neural Network. For each variant, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported between the predicted values and the gold labels of the test set. Best result from all the variations is marked with †.

As can be seen in Table 4.1 and 4.2, for SICK and STS 2017 datasets, GRU based Siamese neural network model outperformed the LSTM based Siamese neural network model which we used as a baseline and this provided the best result. It can be seen that complex architectures that involves Bi-directional RNNs, Attention and Capsule mechanisms did not perform well compared to the simple architectures like GRU. We can conclude that for the smaller datasets like STS 2017 and SICK, GRU based architecture performs better because GRU has less pa-

[2]Pretrained Word2vec can be downloaded from `https://code.google.com/archive/p/word2vec/`

Model	ρ	τ
LSTM	0.831	0.762
Bi-LSTM	0.784	0.708
GRU	0.853†	0.811†
Bi-GRU	0.844	0.804
LSTM + Attention	0.830	0.791
GRU + Attention	0.825	0.782
GRU + Capsule + Flatten	0.806	0.765

Table 4.2: Results for STS 2017 dataset with different variants of Siamese Neural Network. For each variant, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported between the predicted values and the gold labels of the test set. Best result from all the variations is marked with †.

Model	RMSE
LSTM	0.412
Bi-LSTM	0.402
GRU	0.415
Bi-GRU	0.408
LSTM + Attention	0.382†
GRU + Attention	0.398
GRU + Capsule + Flatten	0.421

Table 4.3: Results for QUORA dataset with different variants of Siamese Neural Network. For each variant, Root Mean Squared Error (RMSE) reported between the predicted values and the gold labels of the test set. Best result from all the variations is marked with †.

rameters than LSTM [74]. With less parameters, the architecture does not need a lot of training instances to optimise the training process.

However, when it comes to the big STS dataset; QUORA, the way that the variants of the Siamese neural network behaves is different. As we introduced in Chapter , QUORA was the biggest STS dataset we experimented which has 320,000 training instances. As a result, even the complex architectures like RNNs with Attention get the opportunity to optimise their parameters and deliver good results. This can be seen in Table 4.3. For the QUORA dataset, LSTM + Attention based Siamese neural network model outperformed the LSTM based Siamese neural network model which we used as a baseline and this provided the best result. For bigger datasets, we can conclude that Siamese neural networks based on LSTM with Attention would outperform Siamese neural networks only with LSTMs.

From the experimented variants, one notable observation is the poor performance of capsules in Siamese architectures. Despite providing good results in many NLP tasks like text classification [122, 123] capsule based variant fails to outperform the simple LSTM based variant even in the bigger STS dataset. This implies that capsule based Siamese neural networks won' be a good fit for STS tasks.

With these findings we answer our **RQ1** in this chapter. We have improved the state-of-the-art Siamese neural network architecture and propose a GRU based Siamese neural network architecture for the smaller STS datasets and LSTM+Attention based Siamese neural network for larger STS datasets.

4.3.1 Impact of Transfer Learning

Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. It is a popular approach in deep learning where pre-trained models are used as the starting point for a new task. This is usually done in the scenarios where there is not enough data to train a neural network so that starting from already tuned weights would be advantageous [124, 125]. Transfer learning has often provided good results for smaller datasets. Therefore, we explored the impact of transfer learning, with Siamese neural networks in STS.

We saved the weights of the models that were trained on each STS dataset; SICK, STS 2017 and QUORA. We specifically used the two models that performed best in these dataset; Siamese neural network with GRU and Siamese neural network with LSTM + Attention. We again initiated training for each dataset, however rather than training from scratch, we used the weights of the models trained on other STS dataset. We compared this transfer learning results to the results we got from training the model from scratch. We conducted this transfer learning experiment only on STS2017 and SICK dataset since the QUORA dataset was already big and transfer learning from a smaller dataset to a larger dataset won't make much sense.

Start Model	STS2017	SICK
$STS2017_{GRU}$	0.853	(+0.01)
$STS2017_{LSTM+Aten}$	0.830	(+0.01)
$SICK_{GRU}$	(+0.01)	0.838
$SICK_{LSTM+Aten}$	(+0.01)	0.827
$QUORA_{GRU}$	(-0.02)	(-0.02)
$QUORA_{LSTM+Aten}$	(-0.04)	(-0.04)

Table 4.4: Results for transfer learning with different variants of Siamese Neural Network. For each transfer learning experiment we show the difference between with transfer learning and without transfer learning. Non-grey values are the results of the experiments without transfer learning which we showed in the previous section too. We only report the Pearson correlation due to ease of visualisation.

As can be seen in Table 4.4 some of the transfer learning experiments improved the results for STS2017 and SICK datasets with both architectures. When we performed transfer learning from STS2017 $\Rightarrow$ SICK and SICK $\Rightarrow$ STS2017 the results improve. This shows that transfer learning can improve the results in Siamese neural networks. However, when we performed transfer learning from QUORA $\Rightarrow$ STS2017 and QUORA $\Rightarrow$ SICK the results did not improve, in fact, they decrease, despite QUORA being the largest STS dataset we experimented. This finding somewhat controversial to the general belief in the community that transfer learning from a larger dataset improves the result. In this case, we believe that this happens due to the fact that the QUORA dataset is very different to the other two datasets as we discussed in Chapter 1. Despite QUORA having a large number of training instances, when performing transfer learning, the neural network finds it difficult to optimise the weights for STS2017 and SICK that were already optimised for a very different dataset; QUORA. This result in a decrease in the result. On the other hand transfer learning between STS2017 and SICK improved the results for both datasets since they are similar in nature as we discussed in Chapter 1.

Therefore, we can conclude that transfer learning can improve the results for Siamese neural networks in STS. However, the transfer learning dataset should be picked carefully considering the similarity of the two datasets too, rather than only considering the size of the dataset.

4.3.2 Impact of Data Augmentation

As we observed before, the neural networks perform better when there is a large number of training instances. Therefore, many approaches have been taken to increase the number of training instances. Usually this has resulted better performance with neural networks [126]. Therefore, we experimented the impact of data augmentation with the Siamese neural network architectures we proposed before. We only conducted this experiment with STS 2017 and SICK datasets as QUORA already has a large number of training instances.

We employed thesaurus-based augmentation in which 10,000 additional training examples are generated by replacing random words with one of their synonyms found in Wordnet [127]. A similar approach has been successfully adopted by Mueller and Thyagarajan [117], Zhang et al. [128] too. We specifically used the two models that performed best with the bigger dataset and smaller dataset; Siamese neural network with GRU and Siamese neural network with LSTM + Attention. Since the transfer learning improved the results in previous experiment, we trained the augmented training set on the transferred models; models trained on STS2017 for SICK experiments and models trained on SICK for STS2017.

As can be seen in the results, data augmentation improved the results of all the experiments. However, even with the additional 10,000 training instances, GRU based Siamese neural network outperformed LSTM + Attention based Siamese neural network. We can conclude that simple data augmentation techniques can improves the performance of Siamese neural networks in STS task. From

Dataset	**Start Model**	ρ
SICK	$STS2017_{GRU}$	(+0.01)
	$STS2017_{LSTM+Aten}$	(+0.01)
STS2017	$SICK_{GRU}$	(+0.01)
	$SICK_{LSTM+Aten}$	(+0.01)

Table 4.5: Results for data augmentation with different variants of Siamese Neural Network. For each data augmentation experiment we show the difference between with dat augmentation and without data augmentation. We only report the Pearson correlation (ρ) due to ease of visualisation.

the Siamese neural network experiments we conducted, our best result for both STS2017 and SICK datasets were provided by GRU based Siamese neural network when combined with both transfer learning and data augmentation.

This answers our *RQ2* in this Chapter, we can use transfer learning and simple data augmentation techniques to improve the results of Siamese neural networks in STS.

Model	ρ
Jimenez et al. [129]	0.807
Bjerva et al. [130]	0.827
Zhao et al. [131]	0.841
Siamese LSTM	0.863
Siamese GRU	0.882

Table 4.6: Results for SICK dataset with different variants of Siamese Neural Network. For each variant, Pearson Correlation (ρ) is reported between the predicted values and the gold labels of the test set.

Furthermore, we compared the results of the best Siamese neural network variant with the best results submitted to the competitions [7, 14] and with the unsupervised STS methods we have experimented so far in this part of the thesis. As can be seen in Table 4.6 and 4.7 GRU based Siamese neural network ar-

Model	ρ
Tian et al. [61]	0.851
Siamese LSTM	0.852
Maharjan et al. [132]	0.854
Cer et al. [14]	0.855
Siamese GRU	0.862

Table 4.7: Results for STS2017 dataset with different variants of Siamese Neural Network. For each variant, Pearson Correlation (ρ) is reported between the predicted values and the gold labels of the test set.

chitecture outperforms the best system submitted to both competition. It also outperforms the unsupervised STS methods we have so far explored in this part of the thesis. Therefore, we can conclude that Siamese architecture is currently the best system we have experimented so far for English STS.

4.4 Portability to Other Languages

Our *RQ3* targets the multilinguality aspect of the proposed approach; *Can the proposed Siamese neural network be easily adopted in to different languages?*. To answer this, we evaluated our method in Arabic STS and Spanish STS datasets that were introduced in Chapter 1. Our approach has the advantage that it does not rely on language dependent features. As a result, the approach is easily portable to other languages given the availability of pre-trained word embedding models in that particular language. As word embedding models we used AraVec [57] [3] for Arabic and Spanish 3B words Word2Vec Embeddings [58][4] for Spanish.

[3]AraVec has been trained on Arabic Wikipedia articles. The models are available on `https://github.com/bakrianoo/aravec`

[4]Spanish 3B words Word2Vec Embeddings have been trained on Spanish news articles, Wikipedia articles and Spanish Boletín Oficial del Estado (BOE; English: Official State Gazette). The model is available on `https://github.com/aitoralmeida/spanish_word2vec`

Model	ρ	τ
LSTM	0.746	0.690
Bi-LSTM	0.725	0.683
GRU	$0.763^{\dagger}$	$0.723^{\dagger}$
Bi-GRU	0.752	0.717
LSTM + Attention	0.741	0.703
GRU + Attention	0.739	0.691
GRU + Capsule + Flatten	0.712	0.679

Table 4.8: Results for Arabic STS dataset with different variants of Siamese Neural Network. For each variant, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported between the predicted values and the gold labels of the test set. Best result from all the variations is marked with †.

Model	ρ	τ
LSTM	0.842	0.773
Bi-LSTM	0.814	0.782
GRU	$0.863^{\dagger}$	$0.822^{\dagger}$
Bi-GRU	0.851	0.813
LSTM + Attention	0.845	0.801
GRU + Attention	0.832	0.790
GRU + Capsule + Flatten	0.795	0.773

Table 4.9: Results for Spanish STS dataset with different variants of Siamese Neural Network. For each variant, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported between the predicted values and the gold labels of the test set. Best result from all the variations is marked with †.

As can be seen in Tables 4.9 and 4.8 GRU based Siamese neural network outperformed all the other variants we experimented in both Arabic and Spanish. As we discussed in Chapter 1, both Arabic and Spanish STS datasets we considered are small in size similar to the English STS2017 and SICK datasets. Therefore, similar to STS2017 and SICK datasets, GRU outperform other architecture as GRU does not need a lot of training instances to optimise its weights. It should be noted that it is very easy to adopt this STS method in a different language. We only changed the embeddings to the new language and performed the training.

Furthermore, we compared the results of the best Siamese neural network variant with the best results submitted to the competition [14] and with the unsupervised STS methods we have experimented so far in this part of the thesis.

Model	ρ
Tian et al. [61]	0.744
Nagoudi et al. [133]	0.746
Siamese LSTM	0.746
Wu et al. [59]	0.754
Siamese GRU	0.763

Table 4.10: Results for Arabic STS dataset with different variants of Siamese Neural Network. For each variant, Pearson Correlation (ρ) is reported between the predicted values and the gold labels of the test set.

Model	ρ
Siamese LSTM	0.842
Hassan et al. [134]	0.848
Wu et al. [59]	0.850
Tian et al. [61]	0.855
Siamese GRU	0.863

Table 4.11: Results for Spanish STS dataset with different variants of Siamese Neural Network. For each variant, Pearson Correlation (ρ) is reported between the predicted values and the gold labels of the test set.

As can be seen in Table 4.10 and 4.11 Siamese neural network based on GRU outperforms the top three systems of the competition in both languages. Furthermore, it outperforms the unsupervised STS methods we have experimented with so far in this part of the Thesis. Therefore, we can conclude that Siamese neural network based on GRU is currently the best system we have experimented so far for Arabic and Spanish too.

This answers our **RQ3:**, the Siamese architectures that we propose in this chapter, can be successfully adopted in different language by changing the word embeddings and the training dataset.

4.5 Portability to Other Domains

In order to answer our *RQ4*; how well the proposed Siamese neural network architecture can be applied in different domains, we evaluated our method on Bio-medical STS dataset explained in 1 (BIOSSES). As we mentioned before Bio-medical STS dataset does not have a training set. Therefore, we had to follow a transfer learning strategy to evaluate it on the Bio-medical STS dataset. We used the pre-trained English STS models and performed inference on the Bio-medical STS dataset. We can call it as a *"zero-shot transfer learning"* since the pre-trained English STS models did not see any Bio-medical data.

For this transfer learning strategy we considered two word embedding model; the general Word2vec model we used before [54] that were pre-trained on Google news corpus and BioWordVec [63] which has trained word2vec on a combination of PubMed and PMC texts[5]. With each word embedding model, we trained a Siamese neural network based on GRU and a Siamese neural network based on LSTM + Attention (The two best models we had on English experiments) and evaluated them on the BIOSSES dataset.

As you can see in the Table 4.12 Siamese neural architecture provided sat-

[5]The model is availble on `https://bio.nlplab.org/`

Model	Word2vec	BioWordVec
$STS2017_{GRU}$	0.651	0.721
$STS2017_{LSTM+Aten}$	0.612	0.701
$SICK_{GRU}$	0.642	0.719
$SICK_{LSTM+Aten}$	0.608	0.699
$QUORA_{GRU}$	0.591	0.622
$QUORA_{LSTM+Aten}$	0.603	0.634

Table 4.12: Results for transfer learning with different variants of Siamese Neural Network in BIOSSES dataset. Two considered word embedding models are Word2vec and BioWordVec. We only report the Pearson correlation due to ease of visualisation.

isfactory results. We got the best result from Siamese neural network based on GRU when trained on STS 2017 using BioWordVec. However, the results from SICK dataset is also not far behind. There was a clear improvement when the English STS model was trained using BioWordVec rather than using general Word2vec embeddings. This can be due to the fact that most of the Bio-medical words that appear in BIOSSES dataset are out of vocabulary in general Word2vec embeddings which can cause problems to the neural network when it observes them in the testing phase. Furthermore, it should be noted that in this experiment too, when we performed transfer learning from QUORA dataset the results are lower than performing transfer learning from SICK or STS 2017. This again can be due to the reason SICK and STS 2017 datasets have a similar annotation strategy to the BIOSSES dataset as we discussed in Chapter 1. Even though, QUORA has a large number of training instances, it can't produce good transfer learning results because its annotation strategy is different.

Furthermore we compared our results with the best results reported for the dataset. The results are shown in Table 4.13.

Model	ρ
ELMo $\bigoplus$ *BERT*	0.708
$STS2017_{GRU}$	0.719
Soğancıoğlu et al. [23]	0.754
BioSentVec [79]	0.810

Table 4.13: Results for BIOSSES dataset with different variants of Siamese Neural Network compared with top results reported for BIOSSES. For each variant, Pearson Correlation (ρ) is reported between the predicted values and the gold labels of the test set.

As shown in the results, our method provides satisfactory results when compared with best approaches. However, it should be noted that the unsupervised method we experimented in the previous chapter with BioSentVec [79] comfortably outperformed Siamese neural network approaches we explored in this chapter. We can answer our **RQ4: How well the proposed Siamese neural network perform in a different domain?** with these findings. The Siamese neural network architectures can be adopted to different domains by changing the pre-trained word embeddings. However, without a proper training set the results won't be strong.

4.6 Conclusions

This chapter experimented Siamese neural networks for calculating semantic similarity between pairs of texts and compared them with other unsupervised/ supervised approaches. We used an existing Siamese neural network as the baseline; MALSTM [117] and explored six different variants of Siamese neural networks. We experimented with three English STS datasets, SICK, STS2017 and QUORA. For the smaller STS datasets; SICK and STS2017 we show that Siamese neural network based on GRU outperforms the baseline and for the larger STS dataset, QUORA we show that Siamese neural network with LSTM and Attention outperforms the baseline. Also, we show that we can improve the results more with transfer learning and data augmentation techniques. However, we experienced that performing transfer learning from a bigger dataset won't always improve the results. The quality of the dataset which was used for transfer learning matter too. We show that Siamese neural network based on GRU performs better than the top submissions in both SemEval 2017 task 1 [14] and SemEval 2014 task 1 [7]. The data augmentation techniques we used in this chapter are language dependent as they rely on WordNet [127]. However, as future work we can consider data augmentation techniques that are not language dependant and relies on word embeddings by itself [135].

We extended the experiments with Siamese neural network architectures to Arabic and Spanish STS datasets in SemEval 2017 [14]. In them too the GRU based Siamese neural network architecture outperformed all the systems submitted to the shared task and also outperformed all the STS methods we have explored so far in this part of the thesis. This proves that the Siamese neural network that we propose here can be adopted in different languages. Furthermore, we performed experiments with the BIOSSES dataset. However since the BIOSSES dataset does not have a training set, we had to use transfer learning based zero-shot learning when we are applying Siamese neural networks to this dataset. Even though they provided satisfactory results, Siamese neural networks could not outperform the sentence vector based method we explored in Chapter

2. We can conclude that despite the fact that the Siamese neural networks can be adopted in different domains by changing the word embedding model, they won't provide strong results without a proper training set.

Since word embedding model are now available in most of the languages including the low resource languages like Urdu [136], Telugu [137] and domains like legal domain [138], this method we explored in this chapter can be useful for many languages and domains. However, one drawback is that the need for STS training data in each language and domain which can be challenging in many scenarios.

As future work, it would be interesting to experiment transfer learning between languages with cross-lingual embeddings like fastText [31] using Siamese neural networks. Such approach will be able to train a STS model on resource rich language like English and project the prediction for other languages using zero-shot transfer learning we experimented here. It would be a potential solution for the training data requirement for the low resource languages.

With the introduction of transformer models like BERT [4], Siamese neural networks has evolved incorporating transformers in their architectures too [139]. We will discuss them in Chapter 5.

Chapter 5

Adopting Transformers for STS

5.1 Introduction

Transformer models adopt a pre-training followed by fine-tuning scheme which means that once pre-trained these models can be fine-tuned to a large number of down-stream NLP tasks like text classification, named entity recognition etc. Transformer models, which we have considered in this chapter use special tokens to obtain a single contiguous sequence for each input sequence. Specifically, the first token is always a special classification token [CLS] and sentence pairs are separated using a special token [SEP]. The final hidden state of [CLS] is used for sentence-level fine-tuning tasks and the final hidden state of each token is used for token-level fine-tuning tasks. The fine-tuning scheme in transformers is usually simple like adding a softmax layer on top of [CLS] token for the text classification tasks. Furthermore, the fine-tuning scheme is very efficient as the parameters in the transformer model are already optimised with the pre-training process. Therefore, transformer models have been extremely popular and successful in many NLP tasks [4].

In this chapter, we explore different transformers models in STS. We address four research questions in this chapter:

RQ1: How well the existing state-of-the-art transformer models perform in STS task?

RQ2: Can the method further improved with transfer learning and data augmentation techniques?

RQ3: Can the transformer model be easily adopted in to different languages?

RQ4: How well the proposed transformer models perform in a different domain?

The main contributions of this chapter are as follows.

1. We propose further enhancements to the architecture using transfer learning and data augmentation.

2. We evaluate how well the transformer models perform in different languages and domains.

3. The code and the pre-trained models are publicly available to the community[1]. We have published the code as a python library [2] and by the time of writing this chapter, it has more than 3,000 downloads from the community.

5.2 Related Work

As we mentioned before, after the introduction of BERT [4], many variants of different transformer models have been proposed by adding minor modifications

[1]The public GitHub repository is available on `https://github.com/tharindudr/STS-Transformers`

[2]The developed python library is available on `https://pypi.org/project/ststransformers/`

to the original BERT transformer. Usually these modifications has resulted in improvements in the fine-tuning scheme for the down-stream NLP tasks. Expecting a different performance for the STS task too, we considered following transformer models for the experiments in this chapter.

BERT [4] proposes a masked language modelling (MLM) objective, where some of the tokens of a input sequence are randomly masked, and the objective is to predict these masked positions taking the corrupted sequence as input. BERT applies a Transformer encoder to attend to bi-directional contexts during pre-training. In addition, BERT uses a next-sentence-prediction (NSP) objective. Given two input sentences, NSP predicts whether the second sentence is the actual next sentence of the first sentence. The NSP objective aims to improve the tasks, such as question answering and natural language inference, which require reasoning over sentence pairs.

RoBERTa [6] makes a few changes to the BERT architecture and achieves substantial improvements. The changes include: (1) Training the model longer with larger batches and more data; (2) Removing the NSP objective; (3) Training on longer sequences; (4) Dynamically changing the masked positions during pre-training.

ALBERT [140] proposes two parameter-reduction techniques (factorised embedding parameterisation and cross-layer parameter sharing) to lower memory consumption and speed up training. Furthermore, ALBERT [140] argues that the NSP objective in BERT lacks difficulty, as the negative examples are created by pairing segments from different documents, this mixes topic prediction and coherence prediction into a single task. ALBERT instead uses a sentence-order prediction (SOP) objective. SOP obtains positive examples by taking out two consecutive segments and negative examples by reversing the order of two consecutive segments from the same document.

ELECTRA Compared to BERT, ELECTRA [141] proposes a more effective pre-training method. Instead of corrupting some positions of inputs with [MASK], ELECTRA replaces some tokens of the inputs with their plausible alternatives sampled from a small generator network. ELECTRA trains a discriminator to predict whether each token in the corrupted input was replaced by the generator or not. The pre-trained discriminator can then be used in downstream tasks for fine-tuning, improving upon the pre-trained representation learned by the generator.

XLNET [5] identifies a key weakness in BERT pre-training. Yang et al. [5] argues that the symbols such as [MASK] that are introduced by BERT during pre-training, causes discrepancy between pre-training and fine-tuning as they never occur in real data. Therefore, XLNET suggests a new auto-regressive method based on permutation language modelling (PLM) [142] without introducing any new symbols.

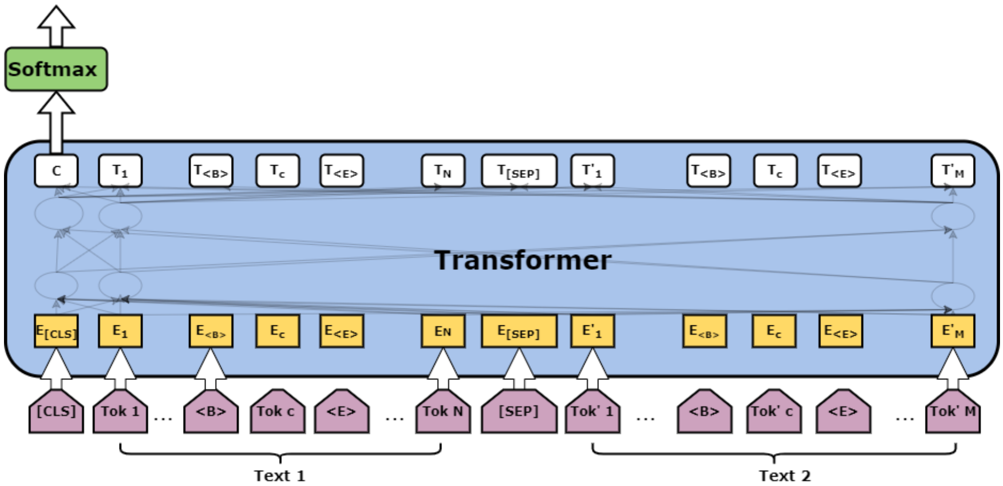


Figure 5.1: Basic architecture for using Transformers in STS.

5.3 Transformer Architecture for STS

5.4 Exploring Transformers in English STS

We evaluated all the above transformer variations in the three English STS datasets we introduced in 1; SICK, STS 2017 and QUORA. We trained the transformer models on the training sets on those datasets and evaluated them on the testing sets. The results are shown in Table 4.1, Table 4.2 and Table 4.3 respectively.

Model	ρ	τ
BERT	0.802	0.733

Table 5.1: Results for SICK dataset with different variants of transformer models. For each variant, Pearson Correlation (ρ) and Spearman Correlation (τ) are reported between the predicted values and the gold labels of the test set. Best result from all the variations is marked with †.

5.4.1 Impact of Transfer Learning

5.4.2 Impact of Data Augmentation

5.5 Portability to Other Languages

5.6 Potability to Other Domains

5.7 Recent Developments: Siamese Transformers

5.8 Conclusions

Part II

Applications - Translation Memories

Chapter 1

Introduction

1.1 What is Translation Memory?

[143]

1.2 Datasets

1.3 Related Work

1.4 STS for Translation Memories

Chapter 2

Sentence Encoders for Translation Memories

2.1 Introduction

[144]

2.2 Methodology

2.3 Results and Evaluation

Part III

Applications - Translation Quality Estimation

Chapter 1

Introduction

1.1 What is Translation Quality Estimation?

1.2 Datasets

1.3 Related Work

[145]

1.4 STS for Translation Quality Estimation

1.5 Conclusion

Chapter 2

TransQuest: STS Architectures for QE

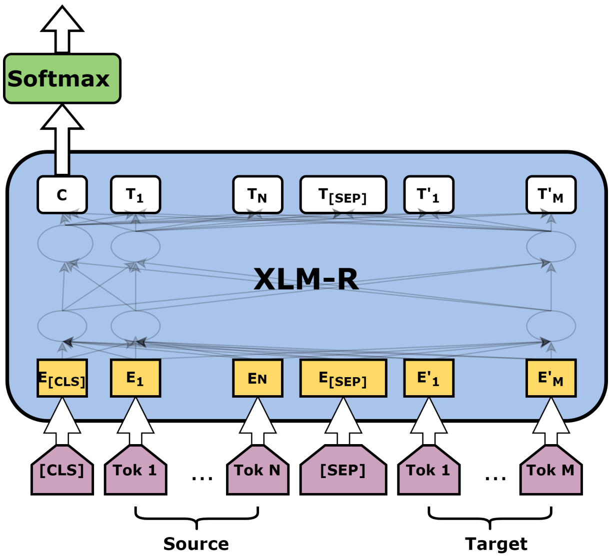
2.1 Introduction

[146]

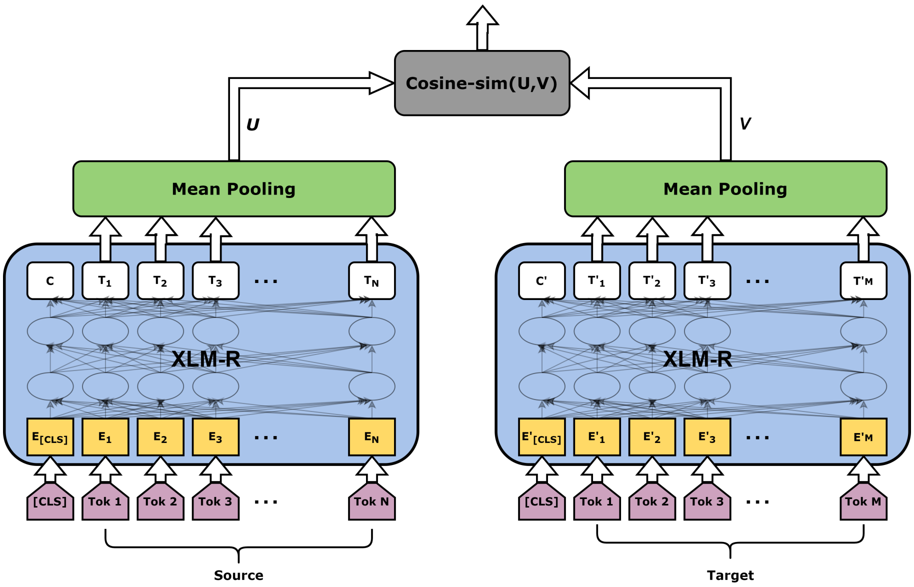
2.2 Methodology

2.3 Results and Evaluation

2.4 Conclusion



(a) *MonoTransQuest* architecture



(b) *SiameseTransQuest* Architecture

Figure 2.1: Architecture in TransQuest

		Mid-resource				High-resource			
	Method	En-Cs SMT	En-Ru NMT	En-Lv SMT	En-Lv NMT	De-En SMT	En-Zh NMT	En-De SMT	En-De NMT
I	MTransQuest	**0.7207**	**0.7126**	0.6592	0.7394	**0.7939**	0.6119	0.7137	**0.5994**
	STransQuest	0.6853	0.6723	0.6320	0.7183	0.7524	0.5821	0.6992	0.5875
II	MTransQuest *-En|En-*	0.7168	0.7046	**0.7181**	**0.7482**	0.7939	0.6101	0.7355	0.5992
	STransQuest *-En|En-*	0.6663	0.6701	0.6533	0.7192	0.7524	0.5721	0.7000	0.5793
III	MTransQuest-m	0.7111	0.7012	0.7141	0.7450	0.7878	0.6092	0.7300	0.5982
	STransQuest-m	0.6561	0.6614	0.6621	0.7202	0.7369	0.5612	0.7015	0.5771
IV	Quest ++	0.3943	0.2601	0.3528	0.4435	0.3323	NR	0.3653	NR
	OpenKiwi	NR	0.5923	NR	NR	NR	0.5058	0.7108	0.4001
	Best system	0.6918	0.5923	0.6188	0.6819	0.7888	**0.6641**	**0.7397**	0.5718
V	mBERT	0.6423	0.6354	0.5772	0.6531	0.7005	0.5483	0.6239	0.5002

Table 2.1: Pearson (r) correlation between *TransQuest* algorithm predictions and human post-editing effort. Best results for each language by any method are marked in bold. Rows I, II and III indicate the different evaluation settings. Row IV shows the results of the state-of-the-art methods and the best system submitted for the language pair in that competition. **NR** implies that a particular result was *not reported* by the organisers. Row V presents the results of the multilingual BERT (mBERT) model in MonoTransQuest Architecture.

Chapter 3

Extending TransQuest for word-level QE

3.1 Introduction

3.2 Related Work

[147]

3.3 Methodology

3.4 Results and Evaluation

3.5 Conclusion

	Train Language(s)	IT En-Cs SMT	IT En-De NMT	IT En-De SMT	IT En-Ru NMT	Pharmaceutical De-En SMT	Pharmaceutical En-LV NMT	Pharmaceutical En-Lv SMT	Wiki En-De NMT	Wiki En-Zh NMT
I	En-Cs SMT	0.6081	(-0.09)	(-0.07)	(-0.09)	(-0.15)	(-0.02)	(-0.01)	(-0.10)	(-0.11)
	En-De NMT	(-0.17)	0.4421	(-0.06)	(-0.02)	(-0.18)	(-0.01)	(-0.02)	(-0.01)	(-0.08)
	En-De SMT	(-0.01)	(-0.05)	0.6348	(-0.67)	(-0.14)	(-0.06)	(-0.04)	(-0.06)	(-0.09)
	En-Ru NMT	(-0.14)	(-0.08)	(-0.16)	**0.5592**	(-0.12)	(-0.01)	(-0.03)	(-0.09)	(-0.08)
	De-En SMT	(-0.43)	(-0.23)	(-0.33)	(-0.31)	**0.6485**	(-0.29)	(-0.32)	(-0.25)	(-0.28)
	En-LV NMT	(-0.12)	(-0.09)	(-0.14)	(-0.03)	(-0.12)	0.5868	(-0.01)	(0.09)	(-0.08)
	En-Lv SMT	(-0.04)	(-0.16)	(-0.10)	(-0.09)	(-0.16)	(-0.01)	0.5939	(-0.15)	(-0.14)
	En-De NMT	(-0.11)	(-0.01)	(-0.08)	(-0.02)	(-0.14)	(-0.02)	(-0.04)	0.6013	(-0.06)
	En-Zh NMT	(-0.19)	(-0.08)	(-0.17)	(-0.03)	(-0.16)	(-0.03)	(-0.06)	(-0.07)	0.6402
II	All	**0.6112**	**0.4523**	**0.6583**	0.5558	0.6221	**0.5991**	**0.5980**	0.6101	0.6229
	All-1	(-0.01)	(-0.01)	(-0.05)	(-0.02)	(-0.12)	(-0.01)	(-0.01)	(-0.01)	(-0.05)
III	Domain	0.6095	0.4467	0.6421	0.5560	0.6331	0.5892	0.5951	0.6021	0.6210
IV	SMT/NMT	0.6092	0.4461	0.6410	0.5421	0.6320	0.5885	0.5934	0.6010	0.6205
V	Baseline-Marmot	0.4449	0.1812	0.3630	NR	0.4373	0.4208	0.3445	NR	NR
	Baseline-OpenKiwi	NR	NR	NR	0.2412	NR	NR	NR	0.4111	0.5583
	Best system	0.4449	0.4361	0.6246	0.4780	0.6012	0.4293	0.3618	**0.6186**	**0.6415**

Table 3.1: Target F1-Multi between the algorithm predictions and human annotations. Best results for each language by any method are marked in bold. Sections I, II and III indicate the different evaluation settings. Section IV shows the results of the state-of-the-art methods and the best system submitted for the language pair in that competition. **NR** implies that a particular result was *not reported* by the organisers. Zero-shot results are colored in grey and it shows the difference between the best result in that section for that language pair and itself.

Chapter 4

Multilingual Quality Estimation with TransQuest

4.1 Introduction

Machine translation quality estimation is generally framed as a supervised machine learning problem [145, 148] where the machine learning models are trained on language specific data for quality estimation. We refer to these models as bilingual QE models. This process would require having annotated QE data for all the language pairs. Furthermore, this language specific supervised machine learning process would result in each machine learning model for each language pair.

This traditional approach has obvious drawbacks. As we mentioned before this process requires training data for each language pair. However, the training data publicly available to build word-level QE models is limited to very few language pairs, which makes it difficult to build QE models for many languages. Furthermore, from an application perspective, even for the languages with resources, it is difficult to maintain separate QE models for each language since the state-of-the-art neural QE models are large in size [149].

To understand the scale of this, consider a real-word application where it is required to build a quality estimation solution for European Parliament. European Parliament has 24 languages which would result in 24*23 language pairs which is equal to 552 language pairs. A traditional bi-lingual QE solution would require 552 training datasets to train the models which is highly challenging and costly to collect and annotate. Furthermore, this would require having 552 machine learning models. State-of-the-art QE models like TransQuest are at least 2GB in size. Having 2GB sized 552 models in the RAM at inferencing time would not be practical. The solution to all these problems is Multilingual QE models.

Multilingual models allow training a single model to perform a task from and/or to multiple languages. Even though this has been applied to many tasks [150, 151] including NMT [152, 153], multilingual approaches have been rarely used in QE [154].

4.2 Multilingual Sentence-Level QE

4.3 Multilingual Word-Level QE

4.4 Conclusion

Bibliography

[1] Hanna Béchara, Hernani Costa, Shiva Taslimipoor, Rohit Gupta, Constantin Orasan, Gloria Corpas Pastor, and Ruslan Mitkov. MiniExperts: An SVM approach for measuring semantic textual similarity. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 96–101, Denver, Colorado, June 2015. Association for Computational Linguistics. doi: 10.18653/v1/S15-2017. URL https://www.aclweb.org/anthology/S15-2017.

[2] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1150. URL https://www.aclweb.org/anthology/P15-1150.

[3] Yang Shao. HCTI at SemEval-2017 task 1: Use convolutional neural network to evaluate semantic textual similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 130–133, Vancouver, Canada, August 2017. Association for Computational Lin-

guistics. doi: 10.18653/v1/S17-2016. URL https://www.aclweb.org/anthology/S17-2016.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

[5] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763, 2019.

[6] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*, 2019.

[7] Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. SemEval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through se-

mantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 1–8, Dublin, Ireland, August 2014. Association for Computational Linguistics. doi: 10.3115/v1/S14-2001. URL https://www.aclweb.org/anthology/S14-2001.

[8] Cyrus Rashtchian, Peter Young, Micah Hodosh, and Julia Hockenmaier. Collecting image annotations using Amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 139–147, Los Angeles, June 2010. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W10-0721.

[9] Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada, 7-8 June 2012. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/S12-1051.

[10] Darnes Vilariño, David Pinto, Saúl León, Mireya Tovar, and Beatriz Beltrán. BUAP: Evaluating features for multilingual and cross-level semantic textual similarity. In *Proceedings of the 8th International Workshop on*

Semantic Evaluation (SemEval 2014), pages 149–153, Dublin, Ireland, August 2014. Association for Computational Linguistics. doi: 10.3115/v1/S14-2022. URL https://www.aclweb.org/anthology/S14-2022.

[11] Rohit Gupta, Hanna Béchara, Ismail El Maarouf, and Constantin Orăsan. UoW: NLP techniques developed at the University of Wolverhampton for semantic similarity and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 785–789, Dublin, Ireland, August 2014. Association for Computational Linguistics. doi: 10.3115/v1/S14-2139. URL https://www.aclweb.org/anthology/S14-2139.

[12] André Lynum, Partha Pakray, Björn Gambäck, and Sergio Jimenez. NTNU: Measuring semantic similarity with sublexical feature representations and soft cardinality. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 448–453, Dublin, Ireland, August 2014. Association for Computational Linguistics. doi: 10.3115/v1/S14-2078. URL https://www.aclweb.org/anthology/S14-2078.

[13] Alexander Chávez, Héctor Dávila, Yoan Gutiérrez, Antonio Fernández-Orquín, Andrés Montoyo, and Rafael Muñoz. UMCC_DLSI_SemSim: Multilingual system for measuring semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 716–721, Dublin, Ireland, August 2014. Association for Computa-

tional Linguistics. doi: 10.3115/v1/S14-2128. URL https://www.aclweb.org/anthology/S14-2128.

[14] Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2001. URL https://www.aclweb.org/anthology/S17-2001.

[15] Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. *SEM 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/S13-1004.

[16] Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. SemEval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 81–91, Dublin, Ireland, August 2014. Association

for Computational Linguistics. doi: 10.3115/v1/S14-2010. URL https://www.aclweb.org/anthology/S14-2010.

[17] Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263, Denver, Colorado, June 2015. Association for Computational Linguistics. doi: 10.18653/v1/S15-2045. URL https://www.aclweb.org/anthology/S15-2045.

[18] Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/S16-1081. URL https://www.aclweb.org/anthology/S16-1081.

[19] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal, September 2015.

Association for Computational Linguistics. doi: 10.18653/v1/D15-1075. URL https://www.aclweb.org/anthology/D15-1075.

[20] Yuxia Wang, Fei Liu, Karin Verspoor, and Timothy Baldwin. Evaluating the Utility of Model Configurations and Data Augmentation on Clinical Semantic Textual Similarity. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing*, pages 105–111, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.bionlp-1.11. URL https://www.aclweb.org/anthology/2020.bionlp-1.11.

[21] Junyi Li, Xuejie Zhang, and Xiaobing Zhou. ALBERT-Based Self-Ensemble Model With Semisupervised Learning and Data Augmentation for Clinical Semantic Textual Similarity Calculation: Algorithm Validation Study. *JMIR Med Inform*, 9(1):e23086, Jan 2021. ISSN 2291-9694. doi: 10.2196/23086. URL http://medinform.jmir.org/2021/1/e23086/.

[22] Z. Imtiaz, M. Umer, M. Ahmad, S. Ullah, G. S. Choi, and A. Mehmood. Duplicate questions pair detection using siamese malstm. *IEEE Access*, 8: 21932–21942, 2020. doi: 10.1109/ACCESS.2020.2969041.

[23] Gizem Soğancıoğlu, Hakime Öztürk, and Arzucan Özgür. BIOSSES: a semantic sentence similarity estimation system for the biomedical domain. *Bioinformatics*, 33(14):i49–i58, 07 2017. ISSN 1367-4803. doi: 10.1093/bioinformatics/btx238. URL https://doi.org/10.1093/bioinformatics/btx238.

[24] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 528–540, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1049. URL https://www.aclweb.org/anthology/N18-1049.

[25] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D17-1070.

[26] Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. From word embeddings to document distances. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 957–966. JMLR.org, 2015.

[27] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page II–1188–II–1196. JMLR.org, 2014.

[28] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=SyK00v5xx.

[29] Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. Enhancing unsupervised sentence similarity methods with deep contextualised word representations. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 994–1003, Varna, Bulgaria, September 2019. INCOMA Ltd. doi: 10.26615/978-954-452-056-4_115. URL https://www.aclweb.org/anthology/R19-1115.

[30] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL http://www.aclweb.org/anthology/D14-1162.

[31] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL https://www.aclweb.org/anthology/L18-1008.

[32] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA, 2013. Curran Associates Inc.

[33] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://www.aclweb.org/anthology/N18-1202.

[34] Ye Jiang, Johann Petrak, Xingyi Song, Kalina Bontcheva, and Diana Maynard. Team bertha von suttner at SemEval-2019 task 4: Hyperpartisan news detection using ELMo sentence representation convolutional network. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 840–844, Minneapolis, Minnesota, USA, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/S19-2146. URL https://www.aclweb.org/anthology/S19-2146.

[35] Ling Luo, Nan Li, Shuaichi Li, Zhihao Yang, and Hongfei Lin. DUTIR at the CCKS-2018 Task1: A neural network ensemble approach for Chinese

clinical named entity recognition. In *CEUR Workshop Proceedings*, volume 2242, pages 7–12. CEUR-WS, 2018.

[36] Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64, Brussels, Belgium, October 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/K18-2005.

[37] Qiao Jin, Bhuwan Dhingra, William Cohen, and Xinghua Lu. Probing biomedical embeddings from language models. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pages 82–89, 2019.

[38] Tharindu Ranasinghe, Marcos Zampieri, and Hansi Hettiarachchi. BRUMS at HASOC 2019: Deep learning models for multilingual hate speech and offensive language identification. In *CEUR Workshop Proceedings*, volume 2517, pages 199–207, 2019. URL http://ceur-ws.org/Vol-2517/T3-3.pdf.

[39] Hansi Hettiarachchi and Tharindu Ranasinghe. Brums at semeval-2020 task 3: Contextualised embeddings for predicting the (graded) effect of context in word similarity. In *Proceedings of the 14th International Work-*

shop on Semantic Evaluation, Barcelona, Spain, 2020. Association for Computational Linguistics.

[40] Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '20, page 1054–1064, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403149. URL https://doi.org/10.1145/3394486.3403149.

[41] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-4013. URL https://www.aclweb.org/anthology/N19-4013.

[42] Wissam Antoun, Fady Baly, and Hazem Hajj. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France, May 2020. European Language Resource Association. ISBN 979-10-95546-51-1. URL https://www.aclweb.org/anthology/2020.osact-1.2.

[43] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.645. URL https://www.aclweb.org/anthology/2020.acl-main.645.

[44] José Cañete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. Spanish Pre-Trained BERT Model and Evaluation Data. In *PML4DC at ICLR 2020*, 2020.

[45] John Koutsikakis, Ilias Chalkidis, Prodromos Malakasiotis, and Ion Androutsopoulos. GREEK-BERT: The Greeks Visiting Sesame Street. In *11th Hellenic Conference on Artificial Intelligence*, SETN 2020, page 110–117, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450388788. doi: 10.1145/3411408.3411440. URL https://doi.org/10.1145/3411408.3411440.

[46] Iz Beltagy, Kyle Lo, and Arman Cohan. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China, November 2019. Association for Computational

Linguistics. doi: 10.18653/v1/D19-1371. URL https://www.aclweb.org/anthology/D19-1371.

[47] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 09 2019. ISSN 1367-4803. doi: 10.1093/bioinformatics/btz682. URL https://doi.org/10.1093/bioinformatics/btz682.

[48] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.261. URL https://www.aclweb.org/anthology/2020.findings-emnlp.261.

[49] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1139.

[50] Alan Akbik, Tanja Bergmann, and Roland Vollgraf. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the 2019*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 724–728, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1078. URL https://www.aclweb.org/anthology/N19-1078.

[51] Shreyas Sharma and Ron Daniel Jr. Bioflair: Pretrained pooled contextualized embeddings for biomedical sequence labeling tasks. *arXiv preprint arXiv:1908.05760*, 2019.

[52] Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-4010. URL https://www.aclweb.org/anthology/N19-4010.

[53] 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, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://www.aclweb.org/anthology/D14-1162.

[54] Tomás, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013. URL http://arxiv.org/abs/1301.3781.

[55] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 604–613, New York, NY, USA, 1998. Association for Computing Machinery. ISBN 0897919629. doi: 10.1145/276698.276876. URL https://doi.org/10.1145/276698.276876.

[56] Murhaf Fares, Andrey Kutuzov, Stephan Oepen, and Erik Velldal. Word vectors, reuse, and replicability: Towards a community repository of large-text resources. In *Proceedings of the 21st Nordic Conference on Computational Linguistics*, pages 271–276, Gothenburg, Sweden, May 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W17-0237.

[57] Abu Bakr Soliman, Kareem Eissa, and Samhaa R. El-Beltagy. Aravec: A set of arabic word embedding models for use in arabic nlp. *Procedia Computer Science*, 117:256–265, 2017. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2017.10.117. URL https://www.sciencedirect.com/science/article/pii/S1877050917321749.

[58] Aritz Bilbao-Jayo and Aitor Almeida. Automatic political discourse analysis with multi-scale convolutional neural networks and contextual data. *International Journal of Distributed Sensor Networks*, 14(11): 1550147718811827, 2018. doi: 10.1177/1550147718811827. URL https://doi.org/10.1177/1550147718811827.

[59] Hao Wu, Heyan Huang, Ping Jian, Yuhang Guo, and Chao Su. BIT at SemEval-2017 task 1: Using semantic information space to evaluate semantic textual similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 77–84, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2007. URL https://www.aclweb.org/anthology/S17-2007.

[60] Sarah Kohail, Amr Rekaby Salama, and Chris Biemann. STS-UHH at SemEval-2017 task 1: Scoring semantic textual similarity using supervised and unsupervised ensemble. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 175–179, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2025. URL https://www.aclweb.org/anthology/S17-2025.

[61] Junfeng Tian, Zhiheng Zhou, Man Lan, and Yuanbin Wu. ECNU at SemEval-2017 task 1: Leverage kernel-based traditional NLP features and neural networks to build a universal model for multilingual and cross-lingual semantic textual similarity. In *Proceedings of the 11th International*

Workshop on Semantic Evaluation (SemEval-2017), pages 191–197, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2028. URL https://www.aclweb.org/anthology/S17-2028.

[62] Joe Barrow and Denis Peskov. UMDeep at SemEval-2017 task 1: End-to-end shared weight LSTM model for semantic textual similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 180–184, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2026. URL https://www.aclweb.org/anthology/S17-2026.

[63] Yijia Zhang, Qingyu Chen, Zhihao Yang, Hongfei Lin, and Zhiyong Lu. Biowordvec, improving biomedical word embeddings with subword information and mesh. *Scientific Data*, 6(1):52, May 2019. doi: 10.1038/s41597-019-0055-0. URL https://doi.org/10.1038/s41597-019-0055-0.

[64] A. R. Aronson. Effective mapping of biomedical text to the umls metathesaurus: the metamap program. *Proceedings. AMIA Symposium*, pages 17–21, 2001. ISSN 1531-605X. URL https://pubmed.ncbi.nlm.nih.gov/11825149. 11825149[pmid].

[65] Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. IndicNLPSuite:

Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.445. URL https://www.aclweb.org/anthology/2020.findings-emnlp.445.

[66] Avihay Chriqui and Inbal Yahav. Hebert & hebemo: a hebrew bert model and a tool for polarity analysis and emotion recognition. *arXiv preprint arXiv:2102.01909*, 2021.

[67] Jesujoba Alabi, Kwabena Amponsah-Kaakyire, David Adelani, and Cristina España-Bonet. Massive vs. curated embeddings for low-resourced languages: the case of Yorùbá and Twi. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2754–2762, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://www.aclweb.org/anthology/2020.lrec-1.335.

[68] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020. doi: 10.1162/tacl_a_00300. URL https://www.aclweb.org/anthology/2020.tacl-1.5.

[69] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-

document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[70] Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*, 2021.

[71] Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder, 2018.

[72] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1188–1196, Bejing, China, 22–24 Jun 2014. PMLR. URL http://proceedings.mlr.press/v32/le14.html.

[73] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

[74] Junyoung Chung, Çaglar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning*, Dec 2014.

[75] M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997. doi: 10.1109/78.650093.

[76] Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1269. URL https://aclanthology.org/D18-1269.

[77] Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-demos.12. URL https://aclanthology.org/2020.acl-demos.12.

[78] Noha S. Tawfik and Marco R. Spruit. Evaluating sentence representations for biomedical text: Methods and experimental results. *Journal of Biomedical Informatics*, 104:103396, 2020. ISSN 1532-0464. doi: https:

//doi.org/10.1016/j.jbi.2020.103396. URL https://www.sciencedirect.com/science/article/pii/S1532046420300253.

[79] Qingyu Chen, Yifan Peng, and Zhiyong Lu. Biosentvec: creating sentence embeddings for biomedical texts. In *2019 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 1–5, 2019. doi: 10.1109/ICHI.2019.8904728.

[80] Mikel Artetxe and Holger Schwenk. Massively Multilingual Sentence Embeddings for Zero-Shot Cross-Lingual Transfer and Beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610, 09 2019. ISSN 2307-387X. doi: 10.1162/tacl_a_00288. URL https://doi.org/10.1162/tacl_a_00288.

[81] Roland Thiolliere, Ewan Dunbar, Gabriel Synnaeve, Maarten Versteegh, and Emmanuel Dupoux. A hybrid dynamic time warping-deep neural network architecture for unsupervised acoustic modeling. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[82] Pranay Manocha, Rohan Badlani, Anurag Kumar, Ankit Shah, Benjamin Elizalde, and Bhiksha Raj. Content-based representations of audio using siamese neural networks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3136–3140, 2018. doi: 10.1109/ICASSP.2018.8461524.

[83] Suwon Shon, Ahmed Ali, and James Glass. Mit-qcri arabic dialect identi-

fication system for the 2017 multi-genre broadcast challenge. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 374–380, 2017. doi: 10.1109/ASRU.2017.8268960.

[84] Yichi Zhang, Bryan Pardo, and Zhiyao Duan. Siamese style convolutional neural networks for sound search by vocal imitation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(2):429–441, 2019. doi: 10.1109/TASLP.2018.2868428.

[85] Jan Švec, Luboš Šmídl, and Josef V. Psutka. An analysis of the rnn-based spoken term detection training. In Alexey Karpov, Rodmonga Potapova, and Iosif Mporas, editors, *Speech and Computer*, pages 119–129, Cham, 2017. Springer International Publishing.

[86] Batuhan Gündoğdu, Bolaji Yusuf, and Murat Saraçlar. Joint learning of distance metric and query model for posteriorgram-based keyword search. *IEEE Journal of Selected Topics in Signal Processing*, 11(8):1318–1328, 2017. doi: 10.1109/JSTSP.2017.2762080.

[87] Aditya Siddhant, Preethi Jyothi, and Sriram Ganapathy. Leveraging native language speech for accent identification using deep siamese networks. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 621–628, 2017. doi: 10.1109/ASRU.2017.8268994.

[88] Neil Zeghidour, Gabriel Synnaeve, Nicolas Usunier, and Emmanuel Dupoux. Joint learning of speaker and phonetic similarities with

siamese networks. In *Interspeech 2016*, pages 1295–1299, 2016. doi: 10.21437/Interspeech.2016-811. URL http://dx.doi.org/10.21437/Interspeech.2016-811.

[89] Wei Zheng, Le Yang, Robert J Genco, Jean Wactawski-Wende, Michael Buck, and Yijun Sun. SENSE: Siamese neural network for sequence embedding and alignment-free comparison. *Bioinformatics*, 35(11):1820–1828, 10 2018. ISSN 1367-4803. doi: 10.1093/bioinformatics/bty887. URL https://doi.org/10.1093/bioinformatics/bty887.

[90] Benjamin Szubert, Jennifer E. Cole, Claudia Monaco, and Ignat Drozdov. Structure-preserving visualisation of high dimensional single-cell datasets. *Scientific Reports*, 9(1):8914, Jun 2019. ISSN 2045-2322. doi: 10.1038/s41598-019-45301-0. URL https://doi.org/10.1038/s41598-019-45301-0.

[91] Minji Jeon, Donghyeon Park, Jinhyuk Lee, Hwisang Jeon, Miyoung Ko, Sunkyu Kim, Yonghwa Choi, Aik-Choon Tan, and Jaewoo Kang. ReSimNet: drug response similarity prediction using Siamese neural networks. *Bioinformatics*, 35(24):5249–5256, 05 2019. ISSN 1367-4803. doi: 10.1093/bioinformatics/btz411. URL https://doi.org/10.1093/bioinformatics/btz411.

[92] Zhiyu Sun, Yusen He, Andrey Gritsenko, Amaury Lendasse, and Stephen Baek. Embedded spectral descriptors: learning the point-wise corre-

spondence metric via Siamese neural networks. *Journal of Computational Design and Engineering*, 7(1):18–29, 03 2020. ISSN 2288-5048. doi: 10.1093/jcde/qwaa003. URL https://doi.org/10.1093/jcde/qwaa003.

[93] Pierre Baldi and Yves Chauvin. Neural networks for fingerprint recognition. *Neural Computation*, 5(3):402–418, 1993. doi: 10.1162/neco.1993.5.3.402.

[94] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1, 2005. doi: 10.1109/CVPR.2005.202.

[95] Haiqing He, Min Chen, Ting Chen, and Dajun Li. Matching of remote sensing images with complex background variations via siamese convolutional neural network. *Remote Sensing*, 10(2), 2018. ISSN 2072-4292. doi: 10.3390/rs10020355. URL https://www.mdpi.com/2072-4292/10/2/355.

[96] Nektarios Paisios, Alex Rubinsteyn, and Lakshminarayanan Subramanian. Choosing which clothes to wear confidently: A tool for pattern matching. In *Frontiers of Accessibility for Pervasive Computing*, June 2012.

[97] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z. Li. Deep metric learning for person re-identification. In *2014 22nd International Conference on Pattern Recognition*, pages 34–39, 2014. doi: 10.1109/ICPR.2014.16.

[98] Grégoire Lefebvre and Christophe Garcia. Learning a bag of features based

nonlinear metric for facial similarity. In *2013 10th IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 238–243, 2013. doi: 10.1109/AVSS.2013.6636646.

[99] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[100] Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. Siamese neural network based similarity metric for inertial gesture classification and rejection. In *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, volume 1, pages 1–6, 2015. doi: 10.1109/FG.2015.7163112.

[101] Majeed Kassis, Jumana Nassour, and Jihad El-Sana. Alignment of historical handwritten manuscripts using siamese neural network. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 293–298, 2017. doi: 10.1109/ICDAR.2017.56.

[102] Muhammad Shehzad Hanif. Patch match networks: Improved two-channel and siamese networks for image patch matching. *Pattern Recognition Letters*, 120:54–61, 2019. ISSN 0167-8655. doi: https://doi.org/10.1016/j.patrec.2019.01.005. URL https://www.sciencedirect.com/science/article/pii/S0167865519300054.

[103] Yanbiao Zou, Jinchao Li, Xiangzhi Chen, and Rui Lan. Learning siamese networks for laser vision seam tracking. *J. Opt. Soc. Am. A*, 35(11):1805–1813, Nov 2018. doi: 10.1364/JOSAA.35.001805. URL http://josaa.osa.org/abstract.cfm?URI=josaa-35-11-1805.

[104] Leen De Baets, Chris Develder, Tom Dhaene, and Dirk Deschrijver. Detection of unidentified appliances in non-intrusive load monitoring using siamese neural networks. *International Journal of Electrical Power & Energy Systems*, 104:645–653, 2019. ISSN 0142-0615. doi: https://doi.org/10.1016/j.ijepes.2018.07.026. URL https://www.sciencedirect.com/science/article/pii/S0142061517331253.

[105] L. V. Utkin, V. S. Zaborovsky, and S. G. Popov. Siamese neural network for intelligent information security control in multi-robot systems. *Automatic Control and Computer Sciences*, 51(8):881–887, Dec 2017. ISSN 1558-108X. doi: 10.3103/S0146411617080235. URL https://doi.org/10.3103/S0146411617080235.

[106] L. V. Utkin, Yu. A. Zhuk, and V. S. Zaborovsky. An anomalous behavior detection of a robot system by using a hierarchical siamese neural network. In *2017 XX IEEE International Conference on Soft Computing and Measurements (SCM)*, pages 630–634, 2017. doi: 10.1109/SCM.2017.7970671.

[107] Andy Zeng, Shuran Song, Kuan-Ting Yu, Elliott Donlon, Francois R. Hogan, Maria Bauza, Daolin Ma, Orion Taylor, Melody Liu, Eudald Romo,

Nima Fazeli, Ferran Alet, Nikhil Chavan Dafle, Rachel Holladay, Isabella Morena, Prem Qu Nair, Druck Green, Ian Taylor, Weber Liu, Thomas Funkhouser, and Alberto Rodriguez. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3750–3757, 2018. doi: 10.1109/ICRA.2018.8461044.

[108] Michael Ryoo, Kiyoon Kim, and Hyun Yang. Extreme low resolution activity recognition with multi-siamese embedding learning. In *AAAI Conference on Artificial Intelligence*, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16790.

[109] Xinchen Liu, Wu Liu, Tao Mei, and Huadong Ma. Provid: Progressive and multimodal vehicle reidentification for large-scale urban surveillance. *IEEE Transactions on Multimedia*, 20(3):645–658, 2018. doi: 10.1109/TMM.2017.2751966.

[110] Wu Liu, Cheng Zhang, Huadong Ma, and Shuangqun Li. Learning efficient spatial-temporal gait features with deep learning for human identification. *Neuroinformatics*, 16(3):457–471, Oct 2018. ISSN 1559-0089. doi: 10.1007/s12021-018-9362-4. URL https://doi.org/10.1007/s12021-018-9362-4.

[111] Sangyun Lee and Euntai Kim. Multiple object tracking via feature pyramid

siamese networks. *IEEE Access*, 7:8181–8194, 2019. doi: 10.1109/ACCESS.2018.2889442.

[112] Wen-tau Yih, Kristina Toutanova, John C. Platt, and Christopher Meek. Learning discriminative projections for text similarity measures. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, CoNLL '11, page 247–256, USA, 2011. Association for Computational Linguistics. ISBN 9781932432923.

[113] Vaibhav Kumar, Dhruv Khattar, Siddhartha Gairola, Yash Kumar Lal, and Vasudeva Varma. Identifying clickbait: A multi-strategy approach using neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 1225–1228, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356572. doi: 10.1145/3209978.3210144. URL https://doi.org/10.1145/3209978.3210144.

[114] José-Ángel González, Encarna Segarra, Fernando García-Granada, Emilio Sanchis, and Llu’ıs-F. Hurtado. Siamese hierarchical attention networks for extractive summarization. *Journal of Intelligent & Fuzzy Systems*, 36:4599–4607, 2019. ISSN 1875-8967. doi: 10.3233/JIFS-179011. URL https://doi.org/10.3233/JIFS-179011. 5.

[115] Arpita Das, Harish Yenala, Manoj Chinnakotla, and Manish Shrivastava. Together we stand: Siamese networks for similar question retrieval. In

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 378–387, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1036. URL https://aclanthology.org/P16-1036.

[116] Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. Learning text similarity with Siamese recurrent networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 148–157, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-1617. URL https://aclanthology.org/W16-1617.

[117] Jonas Mueller and Aditya Thyagarajan. Siamese recurrent architectures for learning sentence similarity. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1), Mar 2016. URL https://ojs.aaai.org/index.php/AAAI/article/view/10350.

[118] Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. Semantic textual similarity with Siamese neural networks. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1004–1011, Varna, Bulgaria, September 2019. INCOMA Ltd. doi: 10.26615/978-954-452-056-4_116. URL https://www.aclweb.org/anthology/R19-1116.

[119] Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. Signature ver-

ification using a “siamese” time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 07(04):669–688, 1993. doi: 10.1142/S0218001493000339. URL https://doi.org/10.1142/S0218001493000339.

[120] Vedran Vukotić, Christian Raymond, and Guillaume Gravier. A step beyond local observations with a dialog aware bidirectional GRU network for Spoken Language Understanding. In *Interspeech*, San Francisco, United States, September 2016. URL https://hal.inria.fr/hal-01351733.

[121] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, 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*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[122] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/2cad8fa47bbef282badbb8de5374b894-Paper.pdf.

[123] Hansi Hettiarachchi and Tharindu Ranasinghe. Emoji powered capsule network to detect type and target of offensive posts in social media. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 474–480, Varna, Bulgaria, September 2019. INCOMA Ltd. doi: 10.26615/978-954-452-056-4_056. URL https://aclanthology.org/R19-1056.

[124] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR, 09–15 Jun 2019. URL http://proceedings.mlr.press/v97/houlsby19a.html.

[125] Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-5004. URL https://aclanthology.org/N19-5004.

[126] Jason Wei and Kai Zou. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-*

IJCNLP), pages 6382–6388, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1670. URL https://aclanthology.org/D19-1670.

[127] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995. ISSN 0001-0782. doi: 10.1145/219717.219748. URL https://doi.org/10.1145/219717.219748.

[128] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 649–657, Cambridge, MA, USA, 2015. MIT Press.

[129] Sergio Jimenez, George Dueñas, Julia Baquero, and Alexander Gelbukh. UNAL-NLP: Combining soft cardinality features for semantic textual similarity, relatedness and entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 732–742, Dublin, Ireland, August 2014. Association for Computational Linguistics. doi: 10.3115/v1/S14-2131. URL https://aclanthology.org/S14-2131.

[130] Johannes Bjerva, Johan Bos, Rob van der Goot, and Malvina Nissim. The meaning factory: Formal semantics for recognizing textual entailment and determining semantic similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 642–646, Dublin,

Ireland, August 2014. Association for Computational Linguistics. doi: 10.3115/v1/S14-2114. URL https://aclanthology.org/S14-2114.

[131] Jiang Zhao, Tiantian Zhu, and Man Lan. ECNU: One stone two birds: Ensemble of heterogenous measures for semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 271–277, Dublin, Ireland, August 2014. Association for Computational Linguistics. doi: 10.3115/v1/S14-2044. URL https://aclanthology.org/S14-2044.

[132] Nabin Maharjan, Rajendra Banjade, Dipesh Gautam, Lasang J. Tamang, and Vasile Rus. DT_Team at SemEval-2017 task 1: Semantic similarity using alignments, sentence-level embeddings and Gaussian mixture model output. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 120–124, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2014. URL https://aclanthology.org/S17-2014.

[133] El Moatez Billah Nagoudi, Jérémy Ferrero, and Didier Schwab. LIM-LIG at SemEval-2017 task1: Enhancing the semantic similarity for Arabic sentences with vectors weighting. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 134–138, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2017. URL https://aclanthology.org/S17-2017.

[134] Basma Hassan, Samir AbdelRahman, Reem Bahgat, and Ibrahim Farag. FCICU at SemEval-2017 task 1: Sense-based language independent semantic textual similarity approach. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 125–129, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2015. URL https://aclanthology.org/S17-2015.

[135] Varun Kumar, Ashutosh Choudhary, and Eunah Cho. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, Suzhou, China, December 2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.lifelongnlp-1.3.

[136] Samar Haider. Urdu word embeddings. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL https://aclanthology.org/L18-1155.

[137] Saurav Kumar, Saunack Kumar, Diptesh Kanojia, and Pushpak Bhattacharyya. "a passage to India": Pre-trained word embeddings for Indian languages. In *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 352–357, Marseille, France, May 2020. European Language Resources association. ISBN 979-10-95546-35-1. URL https://aclanthology.org/2020.sltu-1.49.

[138] Ilias Chalkidis and Dimitrios Kampas. Deep learning in law: early adaptation and legal word embeddings trained on large corpora. *Artificial Intelligence and Law*, 27(2):171–198, Jun 2019. ISSN 1572-8382. doi: 10.1007/s10506-018-9238-9. URL https://doi.org/10.1007/s10506-018-9238-9.

[139] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410.

[140] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1eA7AEtvS.

[141] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1xMH1BtvB.

[142] Benigno Uria, Marc-Alexandre Côté, Karol Gregor, Iain Murray, and Hugo

Larochelle. Neural autoregressive distribution estimation. *Journal of Machine Learning Research*, 17(205):1–37, 2016. URL http://jmlr.org/papers/v17/16-272.html.

[143] Peter J. Arthern. Machine translation and computerized terminology systems: A translator's viewpoint. *Translating and the Computer, Proceedings of a Seminar, London 14th November 1978. Amsterdam: North-Holland Publishing Company*, pages 77–108, 1979.

[144] Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. Intelligent translation memory matching and retrieval with sentence encoders. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 175–184, Lisboa, Portugal, November 2020. European Association for Machine Translation. URL https://www.aclweb.org/anthology/2020.eamt-1.19.

[145] Fabio Kepler, Jonay Trénous, Marcos Treviso, Miguel Vera, and André F. T. Martins. OpenKiwi: An open source framework for quality estimation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 117–122, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-3020. URL https://www.aclweb.org/anthology/P19-3020.

[146] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke

Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL https://www.aclweb.org/anthology/2020.acl-main.747.

[147] Varvara Logacheva, Chris Hokamp, and Lucia Specia. MARMOT: A toolkit for translation quality estimation at the word level. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3671–3674, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL https://www.aclweb.org/anthology/L16-1582.

[148] Dongjun Lee. Two-phase cross-lingual language model fine-tuning for machine translation quality estimation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 1024–1028, Online, November 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.wmt-1.118.

[149] Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. TransQuest: Translation quality estimation with cross-lingual transformers. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5070–5081, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi:

10.18653/v1/2020.coling-main.445. URL https://aclanthology.org/2020.coling-main.445.

[150] Tharindu Ranasinghe and Marcos Zampieri. Multilingual offensive language identification with cross-lingual embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5838–5844, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.470. URL https://aclanthology.org/2020.emnlp-main.470.

[151] Tharindu Ranasinghe and Marcos Zampieri. MUDES: Multilingual detection of offensive spans. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 144–152, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-demos.17. URL https://aclanthology.org/2021.naacl-demos.17.

[152] Toan Q. Nguyen and David Chiang. Transfer learning across low-resource, related languages for neural machine translation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 296–301, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing. URL https://www.aclweb.org/anthology/I17-2050.

[153] Roee Aharoni, Melvin Johnson, and Orhan Firat. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1388. URL https://www.aclweb.org/anthology/N19-1388.

[154] Shuo Sun, Marina Fomicheva, Frédéric Blain, Vishrav Chaudhary, Ahmed El-Kishky, Adithya Renduchintala, Francisco Guzmán, and Lucia Specia. An exploratory study on multilingual quality estimation. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 366–377, Suzhou, China, December 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.aacl-main.39.