Part III

Applications - Translation Quality Estimation

Chapter 8

Introduction to Translation Quality Estimation

The goal of quality estimation (QE) is to evaluate the quality of a translation without having access to a reference translation (Specia et al., 2018). High-accuracy QE that can be easily deployed for a number of language pairs is the missing piece in many commercial translation workflows as they have numerous potential uses. They can be employed to select the best translation when several translation engines are available or can inform the end-user about the reliability of automatically translated content. In addition, QE systems can be used to decide whether a translation can be published as it is in a given context, or whether it requires human post-editing before publishing or even translation from scratch by a human (Kepler et al., 2019).

Quality estimation task is different from automatic machine translation evaluation (Barrault et al., 2020). Automatic machine translation evaluation approaches such as BLEU (Papineni et al., 2002) need the reference translation in order to perform the machine translation evaluation. As mentioned before, quality estimation, on the other hand, does not need the reference translation. Therefore, automatic MT evaluation approaches such as BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), LEPOR (Han et al., 2012) and others can not be directly applied in the QE task. As a result, the solutions proposed for QE are completely different from that of automatic machine translation evaluation.

The estimation of translation quality can be done at different levels: word/phrase-level sentence-level and sentence-level (Ive et al., 2018). Word-level QE aims to spot words that need to be reviewed during the post-editing process. It indicates which words from the source have been incorrectly translated in the target and whether the words inserted between these words are correct. Sentence-level QE models provide a single score for each pair of source and target sentences. Sentence-level QE scores help to rank translations that are worth post-editing. Document-level QE, on the other hand, scores or ranks documents according to their quality for fully automated MT usage scenarios (Ive et al., 2018). In recent years, word-level QE and sentence-level QE have been more popular among the community (Specia et al., 2018).

During the past decade, there has been tremendous progress in the field of quality estimation, largely as a result of the QE shared tasks organised annually by the Workshops on Statistical Machine Translation (WMT), more recently called as the Conferences on Machine Translation, since 2012 (Callison-Burch et al., 2012; Bojar et al., 2013; Bojar et al., 2014; Bojar et al., 2015; Bojar et al., 2016; Bojar et al., 2017; Specia et al., 2018; Fonseca et al., 2019; Specia et al., 2020). First, they have provided annotated datasets which can be used to train QE models and to evaluate them. Second, The annotated datasets these shared tasks released each year have led to the development of many open-source QE systems (Specia et al., 2015; Ive et al., 2018; Kepler et al., 2019).

At present neural-based QE methods constitute state-of-the-art in quality estimation. However, these approaches are based on complex neural networks and require resource-intensive training. This resource-intensive nature of these deep learning based frameworks makes it expensive to train QE models. Furthermore, these architectures require a large number of annotated instances for training, making the quality estimation task very difficult for the low-resource language pairs. This nature of the current state-of-the-art QE systems has hindered their popularity in real-world applications.

This part of the thesis addresses these problems by employing simple STS architectures we experimented with in Part I of the thesis in quality estimation. We redefine the QE task as a cross-lingual STS task and show that state-of-the-art STS architectures can be applied in the QE task by changing the input embeddings. As far as we know, this is the first study done on applying neural STS models directly to the QE task.

In Chapter 9, we explore sentence-level QE with STS architectures. We evaluate their performance in recent sentence-level QE datasets, comparing them with open-source QE tools such as OpenKiwi (Kepler et al., 2019) and QuEst++ (Specia et al., 2015). Finally, we propose a new state-of-the-art QE method for sentence-level QE.

In Chapter 10, we expand this idea to word-level QE. We modify the output of the STS architecture to predict word-level translation qualities. We evaluated their performance in recent word-level QE datasets, comparing the results with open-source word-level QE tools such as OpenKiwi (Kepler et al., 2019) and Marmot (Logacheva et al., 2016). We show that in most of the datasets, our simple word-level architecture outperforms other QE tools.

In Chapter 11, for the first time, we explore multilingual QE with state-of-the-art word-level and sentence-level QE methods. Furthermore, we evaluate multilingual QE in different training environments, including zero-shot and few-shot. Our findings in Chapter 11 would be beneficial for low resource languages.

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

1. We propose two STS architectures based on transformers to perform sentence-level QE. These architectures are simpler than the architectures available in OpenKiwi (Kepler et al., 2019) and DeepQuest (Ive et al., 2018). We evaluate them on 15 language pairs in which sentence-level QE data was available, and we show that the two architectures outperform the current state-of-the-art sentence-level QE frameworks such as OpenKiwi (Kepler et al., 2019) and DeepQuest (Ive et al., 2018).

2. We introduce a simple architecture to perform word-level quality estimation that predicts the quality of the words in the source sentence, target sentence and the gaps in the target sentence. We evaluate it on eight different language pairs in which the word-level QE data was available, and we show that the proposed architecture outperforms the current state-of-the-art word-level QE frameworks like Marmot (Logacheva et al., 2016) and OpenKiwi (Kepler et al., 2019).

3. We propose multilingual learning for QE with the proposed architectures for sentence-level and word-level. We show that multilingual models are helpful in low-resource languages where the training data is difficult to find.

4. We provide important resources to the community. The code of each chapter is bundled to an open-source QE framework, and the pre-trained sentence-level and word-level QE models will be freely available to the community. The link to the relevant code and the models will be unveiled in the introduction section of each chapter.

The remainder of this chapter is structured as follows. Section 8.1 explores the various methods that have been employed in sentence-level and word-level QE, including previous research done incorporating STS in the QE task. Section 8.2 discuss the various datasets we used in this part of the thesis. In Section 8.3, we show the main evaluation metrics used for the sentence-level and word-level QE experiments in the following Chapters in Part III of the thesis. The chapter finishes with the conclusions.

8.1 Related Work

Before the neural network era, most of the quality estimation systems such as QuEst (Specia et al., 2013) and QuEst++ (Specia et al., 2015) were heavily dependent on linguistic processing and feature engineering to train traditional machine-learning algorithms including support vector regression and randomised decision trees (Specia et al., 2013). These features can be either extracted from the machine translation system (*glass-box* features) or obtained from the source and translated sentences, as well as external resources, such as monolingual or parallel corpora (*black-box* features) (Specia et al., 2009). For example, QuEst (Specia et al., 2013) has 17 manually crafted features fed in to a support vector regression algorithm. These 17 features consist of glass-box features such as *ratio of number of tokens in source and target segments*, *ratio of percentage of nouns/verbs in the source and target* etc. as well as black-box features such as *global score and relevant features of the SMT system*, *proportion of pruned search graph nodes* etc. In QuEst++, the number of features varies from 80 to 123 depending on the language pair (Specia et al., 2015). QuEst (Specia et al., 2013), QuEst++ (Specia et al., 2015) and Marmot (Logacheva et al., 2016) can be considered as the most popular traditional QE tools. QuEst (Specia et al., 2013) only supports sentence-level QE, Marmot (Logacheva et al., 2016) only supports word-level QE while QuEst++ (Specia et al., 2015) can support both word-level and sentence-level QE. Even though, they provided good results in early days, these traditional approaches are no longer the state-of-the-art. In recent years, neural-based QE systems have consistently topped the leader boards in WMT quality estimation shared tasks (Kepler et al., 2019).

With the increasing popularity of word embeddings (Mikolov et al., 2013a), neural networks based on word embeddings got popular in the QE field. They outperformed traditional QE systems and provided state-of-the-art results. For example, the best-performing system at the WMT 2017 shared task on

QE was POSTECH, which is purely neural and does not rely on feature engineering at all (Kim et al., 2017). POSTECH revolves around an encoder-decoder Recurrent Neural Network (RNN) (referred to as the 'predictor'), stacked with a bidirectional RNN (the 'estimator') that produces quality estimates. In the predictor, an encoder-decoder RNN model predicts words based on their context representations and in the estimator step, there is a bidirectional RNN model to produce quality estimates for words, phrases and sentences based on representations from the predictor. To be effective, POSTECH requires extensive predictor pre-training, which means it depends on large parallel data and is computationally intensive (Ive et al., 2018). The POSTECH architecture was later re-implemented in DeepQuest (Ive et al., 2018). DeepQuest supports both word-level and sentence-level QE (Ive et al., 2018).

OpenKiwi (Kepler et al., 2019) is another open-source neural QE framework developed by Unbabel. It implements four different neural network architectures QUETCH (Kreutzer et al., 2015), NuQE (Martins et al., 2016), Predictor-Estimator (Kim et al., 2017) and a stacked model of these architectures. Both the QUETCH and NuQE architectures have simple neural network models that do not rely on additional parallel data but do not perform that well. The Predictor-Estimator model is similar to the POSTECH architecture and relies on additional parallel data. In OpenKiwi, the best performance for sentence-level quality estimation was given by the stacked model that used the Predictor-Estimator model, meaning that the best model requires extensive predictor pre-training and relies on large parallel data and computational resources.

As discussed before, these models' complex and resource-intensive nature creates many limitations when deployed in real-world scenarios. Therefore, in this study, we propose to use simple STS architectures in QE. Over the years, there have been a few attempts to integrate semantic similarity into QE. Specia et al. (2011) employed semantic information into the QE task to address the problem of meaning preservation in translation. The authors integrated semantic similarity features to the QE model and improved the results of the QE task (Specia et al., 2011). Biçici and Way (2014) introduced the use of referential translation machines (RTM) for QE. RTM is a computational model for judging monolingual and bilingual similarity that achieves state-of-the-art results. This approach provided the best result in both sentence-level and word-level tasks in WMT 2013 (Bojar et al., 2013). Furthermore, Kaljahi et al. (2014) and Souza et al. (2014) used syntactic and semantic information in quality estimation and were able to improve over the baseline when combining these features with the features of the baseline. Finally, in a different approach, Bechara et al. (2016) used semantically similar sentences and their quality scores as features to estimate the quality of machine translated sentences. This method improved the prediction of machine translation quality for semantically similar sentences (Bechara et al., 2016).

Even though there are several studies done to integrate semantic similarity into QE, as far as we know, this is the first study to employ state-of-the-art neural STS models directly in the QE task.

8.2 Datasets

All the datasets that we used in this part of the thesis are publicly available and were released in WMT quality estimation tasks in recent years (Specia et al., 2018; Fonseca et al., 2019; Specia et al., 2020). This was done to ensure the replicability of our experiments and to allow us to compare our results with state-of-the-art methods. The following sections will describe the sentence-level and word-level QE datasets we experimented, separately.

8.2.1 Sentence-level QE

Sentence-level QE datasets that we used in this part of the thesis can be categorised into two main areas depending on the aspect they have been annotated; Human-mediated Translation Edit Rate (HTER) and Direct Assessment (DA). Most of the early datasets have been annotated on HTER. Very recently DA aspect is getting popular in the QE community. We describe each of them in detail in the following sections.

Predicting HTER The performance of sentence-level QE systems has typically been assessed using the semiautomatic HTER. HTER is an edit-distance-based measure that captures the distance between the automatic translation and a reference translation in terms of the number of modifications required to transform one into another. In light of this, a QE system should be able to predict the percentage of edits required in the translation. We used several language pairs for which HTER information was available: English-Chinese (En-

Language Pair	Source	MT system	Competition	train, dev, test size
De-En	Pharmaceutical	Phrase-based SMT	WMT 2018 (Specia et al., 2018)	25,963, 1,000, 1,000
En-Zh	Wiki	fairseq based NMT	WMT 2020 (Specia et al., 2020)	7,000, 1,000, 1,000
En-Cs	IT	Phrase-based SMT	WMT 2018 (Specia et al., 2018)	40,254, 1,000, 1,000
En-De	Wiki	fairseq based NMT	WMT 2020 (Specia et al., 2020)	7,000, 1,000, 1,000
En-De	IT	Phrase-based SMT	WMT 2018 (Specia et al., 2018)	26,273, 1,000, 1,000
En-Ru	Tech	Online NMT	WMT 2019 (Fonseca et al., 2019)	15,089, 1,000, 1,000
En-Lv	Pharmaceutical	Attention-based NMT	WMT 2018 (Specia et al., 2018)	12,936, 1,000, 1,000
En-Lv	Pharmaceutical	Phrase-based SMT	WMT 2018 (Specia et al., 2018)	11,251, 1,000, 1,000

Table 8.1: Information about language pairs used to predict HTER. The **Language Pair** column shows the language pairs we used in ISO 639-1 codes[1]. **Source** expresses the domain of the sentence and **MT system** is the Machine Translation system used to translate the sentences. In that column NMT indicates Neural Machine Translation and SMT indicates Statistical Machine Translation. **Competition** shows the quality estimation competition in which the data was released and the last column indicates the number of instances the train, development and test dataset had in each language pair respectively.

Zh), English-Czech (En-Cs), English-German (En-De), English-Russian (En-Ru), English-Latvian (En-Lv) and German-English (De-En). The texts are from a variety of domains, and the translations were produced using both neural and statistical machine translation systems. More details about these datasets can be found in Table 8.1 and in (Specia et al., 2018; Fonseca et al., 2019; Specia et al., 2020). Several examples from WMT 2020, En-De dataset is shown in Table 8.2.

Predicting DA Even though HTER has been typically used to assess quality in machine translations, the reliability of this metric for evaluating the performance of quality estimation systems has been questioned by researchers (Graham et al., 2016). The current practice in MT evaluation is the so-called Direct Assessment (DA) of MT quality (Graham et al., 2017), where raters evaluate the machine translation on a continuous 1-100 scale. This method has been shown to improve

[1]Language codes are available in ISO 639-1 Registration Authority Website Online - `https://www.loc.gov/standards/iso639-2/php/code_list.php`

Source	Target	HTER
José Ortega y Gasset visited Husserl at Freiburg in 1934 .	1934 besuchte José Ortega y Gasset Husserl in Freiburg .	0.3333
however , a disappointing ninth in China meant that he dropped back to sixth in the standings .	eine enttäuschende Neunte in China bedeutete jedoch , dass er in der Gesamtwertung auf den sechsten Platz zurückfiel .	0.2000
" Renaissance Humanism and the Future of the Humanities . "	" Renaissance Humanism and the Future of the Humanities " .	1.0000
sophomore Jacory Harris directed the newly implemented offense .	Sophomore Jacory Harris leitete die neu umgesetzte Straftat .	0.0000

Table 8.2: Examples source/target pairs from WMT 2020 En-De HTER dataset (Specia et al., 2020). **Source** column shows the source sentence in English and **Target** shows the target sentence in German. **HTER** column shows the annotated HTER value for the translation.

the reproducibility of manual evaluation and to provide a more reliable gold standard for automatic evaluation metrics (Graham et al., 2015).

We used a recently created dataset to predict DA in machine translations which was released for the WMT 2020 quality estimation shared task 1 (Specia et al., 2020). The dataset is composed of data extracted from Wikipedia for six language pairs, consisting of high-resource English-German (En-De) and English-Chinese (En-Zh), medium-resource Romanian-English (Ro-En) and Estonian-English (Et-En), and low-resource Sinhala-English (Si-En) and Nepalese-English (Ne-En), as well as a Russian-English (En-Ru) dataset which combines articles from Wikipedia and Reddit (Fomicheva et al., 2020b). These datasets have been collected by translating sentences sampled from source-language articles using state-of-the-art NMT models built using the fairseq toolkit (Ott et al., 2019) and annotated with DA scores by professional translators.

Source	**Target**	**DA**
Tratatul de Aderare fusese semnat la 16 aprilie 2003.	The Accession Treaty had been signed on 16 April 2003.	0.6226
Are o industrie turistică bine dezvoltată, fiind o destinație populară printre turiștii britanici și germani.	It has a well-developed tourist industry as a popular destination among British and German tourists.	0.9386
Numărul mare de avioane de vânătoare trimise în misiuni împotriva avioanelor de recunoaștere nu a fost o greșeală.	There was no mistake in the large number of hunting aeroplanes deployed in missions against recognition aeroplanes.	0.0024
Ar fi fost inutil să încerce obținerea unei o catedre universitare.	It would have been pointless to try to get a university catalogue.	0.5456

Table 8.3: Examples source/target pairs from WMT 2020 Ro-En DA dataset (Specia et al., 2020). **Source** column shows the source sentence in English and **Target** shows the target sentence in German. **DA** column shows the annotated DA value for the translation.

Each translation was rated with a score from 0-100 according to the perceived translation quality by at least three translators (Specia et al., 2020). The DA scores were standardised using the z-score. The quality estimation systems evaluated on these datasets have to predict the mean DA z-scores of test sentence pairs. Each language pair has 7,000 sentence pairs in the training set, 1,000 sentence pairs in the development set and another 1,000 sentence pairs in the testing set.

8.2.2 Word-level QE

Word-level QE annotations are not straightforward as sentence-level QE annotations. They have been annotated for words in the target (‘OK’ for correct words, ‘BAD’ for incorrect words), gaps in the target (‘OK’ for genuine gaps, ‘BAD’ for gaps indicating missing words) and source words (‘BAD’ for words that lead to errors in the target, 'OK' for other words) (Specia et al., 2018). To make it clearer, consider the following example from En-De in WMT 2018 (Specia et al., 2018). The word-level quality estimation labels would be followed. Please note that *green* represents 'OK' and *red* represents 'BAD' labels.

Source - *for example , you could create a document containing a car that moves across the Stage .*

Target - *Sie können beispielsweise ein Dokument erstellen , das ein Auto über die Bühne enthält .*

Source - *for example , you could create a document containing a car that moves across the Stage .*

Target - *<GAP> Sie <GAP> können <GAP> beispielsweise <GAP> ein <GAP> Dokument <GAP> erstellen <GAP> , <GAP> das <GAP> ein <GAP> Auto <GAP> über <GAP> die <GAP> Bühne <GAP> enthält <GAP> . <GAP>*

As can be seen in the example, all the words in the source, all the words in the target and all the *gaps* in the target have been annotated as 'OK' or 'BAD'. Most of the language pairs that are annotated for sentence-level HTER scores also have word-level quality labels. Therefore, for the word-level QE experiments, we used the same language pairs we used for sentence-level HTER, which are shown in Table 8.1. More information about the word-level annotations are available on (Specia et al., 2018; Fonseca et al., 2019; Specia et al., 2020).

8.3 Evaluation

For the evaluation, we used the same approach proposed in the WMT shared tasks so that we can compare our results with the respective baselines and the best systems submitted in each shared task. Obviously, the sentence-level and word-level QE follow two different evaluation criteria, which we explain in the following sections.

Sentence-Level QE Evaluation : Similar to the STS evaluation in Part I of the thesis, Pearson's Correlation Coefficient (ρ) is the most popular evaluation metric in recent WMT sentence-level QE shared tasks (Specia et al., 2018; Fonseca et al., 2019; Specia et al., 2020). Since the gold labels are continuous in both HTER and DA and the models need to predict a continuous value, it makes sense to employ Pearson's Correlation Coefficient as the evaluation metric. A QE 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 sentence-level QE.

Word-level QE Evaluation : The primary evaluation metric for word-level QE is the multiplication of F1-scores for the OK and BAD classes, denoted as $F1_{MULTI}$ (Specia et al., 2018; Fonseca et al., 2019; Specia et al., 2020). The standard equation for the F1 score is shown in Equation 8.1 where TP, TN, FP and FN are True Positive, True Negative, False Positive and False Negative, respectively. This F1 score is calculated for OK and BAD classes separately, and then, they are multiplied to get the $F1_{MULTI}$ as shown in Equation 8.2.

$$F1 = \frac{2 * TP}{2 * TP + FP + FN} \tag{8.1}$$

$$F1_{MULTI} = F1_{OK} \times F1_{BAD} \tag{8.2}$$

Prior to WMT 2019, $F1_{MULTI}$ score was calculated separately for words in the source ($F1_{MULTI}$ *Source*), words in the target ($F1_{MULTI}$ *Target*) and gaps in the target ($F1_{MULTI}$ *GAPS*) (Specia et al., 2018), while after WMT 2019 (Fonseca et al., 2019; Specia et al., 2020) they produce a single result for target gaps and words named as "$F1_{MULTI}$ *Target*" alongside with "$F1_{MULTI}$ *Source*". We follow the same approach. A QE model with a $F1_{MULTI}$ score close to 1 indicates that the predictions of that model and gold labels are similar and therefore, it is a good model to predict word-level QE.

8.4 Conclusion

Quality estimation is an important component in making machine translation useful in real-world applications. It aims to inform the user of the quality of the MT output at test time. This process can be done on different levels such as word-level, sentence-level and document-level. QE shared tasks organised annually by WMT have increased QE's popularity among the MT community by leading the development of standard datasets covering a variety of languages and domains. These QE shared tasks have further contributed to the development of evaluation measures in QE. We followed the same evaluation measures; Pearson's Correlation Coefficient for sentence-level and $F1_{MULTI}$ for word-level.

The annotated datasets these shared tasks released each year have led to the development of many open-source QE systems. Similar to other NLP fields, most of the early QE approaches including QuEst (Specia et al., 2013), QuEst++ (Specia et al., 2015) and Marmot (Logacheva et al., 2016) were also based on traditional machine learning and involved heavy feature engineering. However, they no longer provide competitive results. The current state-of-the-art in QE is neural models. Following this, many open-source neural QE frameworks such as OpenKiwi (Kepler et al., 2019) and DeepQuest (Ive et al., 2018) have been created. However, these neural QE architectures are complex and need a lot of computing resources to train a QE model, which we have identified as a major limitation. To address this weakness, we propose to redefine the QE task as a cross-lingual STS task and apply the STS architectures we experimented in Part I of the thesis to QE, which are considerably simpler than the current state-of-the-art QE models.

In the next few chapters, we will be exploring the STS architectures in the QE task. First, in Chapter 9, the STS architectures will be applied in sentence-level QE, and then in Chapter 10, these architectures will be extended to word-level QE. Finally, in Chapter 11, we explore multilingual QE with the proposed architectures.

Chapter 9

TransQuest: STS Architectures for QE

Neural-based QE methods constitute state-of-the-art in quality estimation. This is clear as, in recent years, neural-based QE systems have consistently topped the leaderboards in WMT quality estimation shared tasks (Kepler et al., 2019). For example, the best-performing system at the WMT 2017 shared task on QE was POSTECH, which is purely neural and does not rely on feature engineering at all (Kim et al., 2017). As a result, most of the recent open-source QE frameworks such as OpenKiwi (Kepler et al., 2019) and DeepQuest (Ive et al., 2018) rely on deep learning.

However, despite providing state-of-the-art results in QE, these neural frameworks have a common drawback. They are very complex and need a lot of computing resources. For example, the best performing sentence-level neural architecture in OpenKiwi (Kepler et al., 2019) is the stacked model that used the Predictor-Estimator model we described in Chapter 8 which requires extensive predictor pre-training and relies on a large parallel dataset and computational resources. This is similar to the POSTECH architecture in DeepQuest (Ive et al., 2018) . This complex nature of the state-of-the-art QE frameworks has hindered their popularity in real-life applications.

To overcome this, we propose to remodel the QE task as a cross-lingual STS task. In other words, measuring the quality between a source and a target can be interpreted as computing the cross-lingual textual similarity between the source and the target. With this definition, we can use the STS neural architectures we explored in Part I of the thesis in the QE task. However, since we used monolingual embeddings for STS, we need to change them so that the embeddings can represent both languages in the QE task. For that, we propose to use cross-lingual embeddings. We assume that using the cross-lingual embeddings in the same vector space would ease the learning process for the proposed neural network architectures.

Recalling from Part I of the thesis, state-of-the-art supervised STS methods rely on transformer models. These architectures are considerably simpler than state-of-the-art QE architectures in OpenKiwi (Kepler et al., 2019) and DeepQuest (Ive et al., 2018). Furthermore, cross-lingual embeddings that we intend to use with the STS architectures are fine-tuned to reflect the properties between source and target languages. As a result, this removes the dependency on large parallel data, which in turn means there is no longer need for powerful computational resources. These reasons motivated us to explore STS architectures in the QE task. We believe that simpler and efficient architectures will improve the popularity of QE in real-life applications. As far as we know, this would be the first work to apply STS architectures directly in the QE task.

We address three research questions in this chapter:

RQ1: Can existing state-of-the-art STS architecture be used in sentence-level

QE tasks by modifying the input embeddings?

RQ2: Do cross-lingual embeddings have an advantage over multilingual embeddings in the QE task?

RQ3: Can the models be further improved by data augmentation and ensemble learning?

The main contributions of this chapter are as follows.

1. We propose two architectures based on Transformers to perform sentence-level QE. These architectures are simpler than the architectures available in OpenKiwi and DeepQuest (Lee, 2020; Wang et al., 2018).

2. We evaluate them on both aspects of sentence-level QE on 15 language pairs, and we show that the two architectures outperform the current state-of-the-art sentence-level QE frameworks like DeepQuest (Ive et al., 2018) and OpenKiwi (Kepler et al., 2019).

3. We suggest further improvements to the models by data augmentation and ensemble learning.

4. The two architectures introduced here were released as part of a QE framework; *TransQuest.* TransQuest was independently evaluated in the WMT 2020 QE shared task 1[1] (Specia et al., 2020) and won it in all the language pairs outperforming 50 teams around the globe (Ranasinghe et al., 2020b).

[1]The shared task is available on http://statmt.org/wmt20/quality-estimation-task.html

5. The code and the pre-trained models of *TransQuest* are publicly available to the community[2]. We have published *TransQuest* as a python library[3], and by the time of writing this chapter, it has more than 9,000 downloads from the community[4].

The rest of this chapter is organised as follows. Section 9.1 discusses the methodology and the experiments conducted with 15 language pairs in both aspects of sentence-level QE. Section 9.2 shows the results and Section 9.3 provides further fine-tuning strategies to improve the results. In Section 9.4, we provide a basic error analysis. The chapter finishes with conclusions and ideas for future research directions in QE.

9.1 Methodology

As mentioned before, we considered sentence-level QE as a cross-lingual STS task. Therefore, we used the two best STS architectures we had in part I of the thesis. As we mentioned in Part I, these architectures were based on Transformers, the default sentence pair classification architecture in Transformers and the Siamese Transformer model described in Chapter 5. However, rather than using monolingual Transformers as in STS, we used cross-lingual Transformers for the QE task to represent both languages in the same vector space.

[2]The public GitHub repository is available on `https://github.com/tharindudr/TransQuest`. The pre-trained QE models for more than 15 language pairs are available on HuggingFace model repository on `https://huggingface.co/TransQuest`

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

[4]For the latest statistics, please visit `https://pepy.tech/project/transquest`

Although there are several multilingual transformer models including multilingual BERT (mBERT) (Devlin et al., 2019) and multilingual DistilBERT (mDistilBERT) (Sanh et al., 2019), researchers expressed some reservations about their ability to represent all the languages (Pires et al., 2019). In addition, although mBERT and mDistilBERT showed some cross-lingual characteristics, they do not perform well on cross-lingual benchmarks (K et al., 2020) as they have not been trained using cross-lingual data.

XLM-RoBERTa (XML-R) was released in November 2019 (Conneau et al., 2020) as an update to the XLM-100 model (Conneau and Lample, 2019). XLM-R takes a step back from XLM, eschewing XLM's Translation Language Modeling (TLM) objective since it requires a dataset of parallel sentences, which can be difficult to acquire. Instead, XLM-R trains RoBERTa(Liu et al., 2019) on a huge, multilingual dataset at an enormous scale: unlabelled text in 104 languages, extracted from CommonCrawl datasets, totalling 2.5TB of text. It is trained using only RoBERTa's (Liu et al., 2019) masked language modelling (MLM) objective. Surprisingly, this strategy provided better results in cross-lingual tasks. XLM-R outperforms mBERT on a variety of cross-lingual benchmarks such as cross-lingual natural language inference and cross-lingual question-answering (Conneau et al., 2020). This superior performance of XLM-R in cross-lingual tasks motivated us to use XLM-R in QE.

The *TransQuest* framework that is used to implement the two architectures described here relies on the XLM-R transformer model (Conneau et al., 2020) to derive the representations of the input sentences. The XLM-R transformer model takes a sequence of no more than 512 tokens as input and outputs the representation of the sequence. The first token of the sequence is always [CLS], which contains the special embedding to represent the whole sequence, followed by embeddings acquired for each word in the sequence. As shown below, our proposed neural network architectures can utilise both the embedding for the [CLS] token and the embeddings generated for each word. The output of the transformer (or transformers for *SiameseTransQuest* described below) is fed into a simple output layer which is used to estimate the quality of a translation. We describe below how the XLM-R transformer is used and the output layer, as they are different in the two instantiations of the framework. The fact that we do not rely on a complex output layer makes training our architectures much less computationally intensive than state-of-the-art QE solutions such as predictor-estimator (Lee, 2020; Wang et al., 2018).

There are two pre-trained XLM-R models released by HuggingFace's Transformers library (Wolf et al., 2020); XLM-R-base and XLM-R-large. We used both of these pre-trained models in our experiments[5]. Both of these pre-trained XLM-R models cover 104 languages (Conneau et al., 2020), making it potentially very useful to estimate the translation quality for a large number of language pairs.

1. **MonoTransQuest** (**MTransQuest**): The first architecture proposed uses a single XLM-R transformer model and is shown in Figure 9.1a. This

[5]XLM-R-large is available on `https://huggingface.co/xlm-roberta-large` and XLM-R-base is available on `https://huggingface.co/xlm-roberta-base`

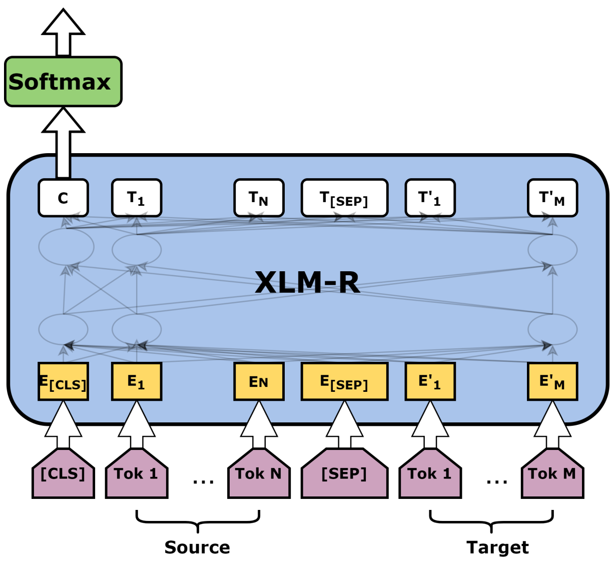
architecture produced the best results for monolingual STS tasks in Part I of the thesis. The input of this model is a concatenation of the original sentence and its translation, separated by the [SEP] token. We experimented with three pooling strategies for the output of the transformer model: using the output of the [CLS] token (`CLS`-strategy), computing the mean of all output vectors of the input words (`MEAN`-strategy), and computing a max-over-time of the output vectors of the input words (`MAX`-strategy). The output of the pooling strategy is used as the input of a softmax layer that predicts the quality score of the translation. We used the mean-squared-error loss as the objective function. Early experiments we carried out demonstrated that the `CLS`-strategy leads to better results than the other two strategies for this architecture. Therefore, we used the embedding of the [CLS] token as the input of a softmax layer.

2. **SiameseTransQuest** (**STransQuest**): The second approach proposed in this chapter relies on the Siamese architecture depicted in Figure 9.1b showed promising results in monolingual STS tasks (Reimers and Gurevych, 2019) in Part I of the thesis. In this case, we feed the original text and the translation into two separate XLM-R transformer models. Similar to the previous architecture, we used the same three pooling strategies for the outputs of the transformer models. We then calculated the cosine similarity between the two outputs of the pooling strategy.

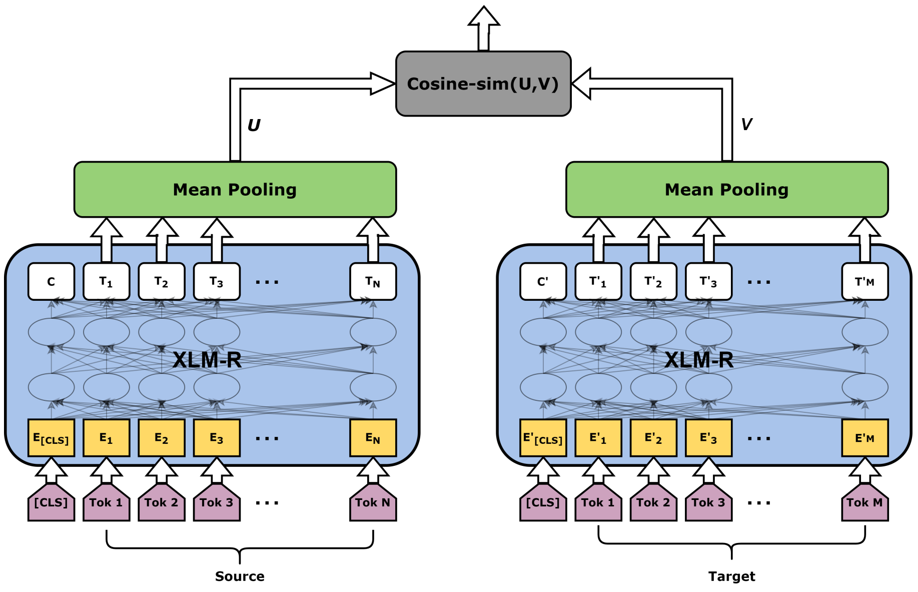
We used the mean-squared-error loss as the objective function. In the initial experiments we carried out with this architecture, the `MEAN`-strategy showed better results than the other two strategies. For this reason, we used the `MEAN`-strategy for our experiments. Therefore, cosine similarity is calculated between the the mean of all output vectors of the input words produced by each transformer.

9.1.1 Experimental Setup

We evaluated these two architectures in both aspects of sentence-level quality estimation using the data introduced in Chapter 8. We applied the same set of configurations for all the language pairs in order to ensure consistency between all the experiments. This also provides a good starting configuration for researchers who intend to use TransQuest on a new language pair. In both architectures, we used a batch-size of eight, Adam optimiser with $2e^{-5}$ learning rate, and a linear learning rate warm-up over 10% of the training data. During the training process, the parameters of the XLM-R model and the subsequent layers were updated. The models were trained using only training data. Furthermore, they were evaluated while training once in every 300 training steps using an evaluation set that had one-fifth of the rows in training data. We performed early stopping if the evaluation loss did not improve over ten evaluation steps. All the models were trained for three epochs.



(a) *MonoTransQuest* Architecture



(b) *SiameseTransQuest* Architecture

Figure 9.1: Two architectures employed in TransQuest.

9.2 Results and Discussion

The results for HTER and DA aspects are shown in Tables 9.1 and 9.2. We report the Pearson correlation (ρ) between the predictions and the gold labels from the test set, which is the most commonly used evaluation metric in recent WMT quality estimation shared tasks (Specia et al., 2018; Fonseca et al., 2019; Specia et al., 2020) as mentioned in Chapter 8. Since we use the same evaluation process, we could compare our results with the baselines and best systems of the respective shared task. Raw I in Tables 9.1 and 9.2 shows the results for *MonoTransQuest* and *SiameseTransQuest* with XLM-R-large and Raw II in Tables 9.1 and 9.2 shows the results for *MonoTransQuest* and *SiameseTransQuest* with XLM-R-base. As can be seen in results, XLM-R-large always outperformed XLM-R-base in both architectures. Therefore, we use the results we got from XLM-R-large for the following analysis.

In the HTER aspect of sentence-level quality estimation, as shown in Table 9.1, *MonoTransQuest* gains $\approx$ 0.1-0.2 Pearson correlation boost over OpenKiwi in most language pairs. In the language pairs where OpenKiwi results are not available, *MonoTransQuest* gains $\approx$ 0.3-0.4 Pearson correlation boost over QuEst++ in all language pairs for both NMT and SMT. Table 9.1 also gives the results of the best system submitted for a particular language pair. It is worth noting that the *MonoTransQuest* results surpass the best system in all the language pairs with the exception of the En-De SMT, En-De NMT and En-Zh NMT datasets. *SiameseTransQuest* falls behind *MonoTransQuest* architecture, however it outperforms OpenKiwi and Quest++ in all the language pairs except En-De SMT. This shows that simple STS models with cross-lingual embeddings outperform the complex state-of-the-art QE models in predicting HTER.

As shown in Table 9.2, in the DA aspect of quality estimation,

		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-base	0.7012	0.6982	0.6254	0.7110	0.7653	0.5873	0.6872	0.5762
	STransQuest-base	0.6678	0.6542	0.6132	0.6892	0.7251	0.5551	0.6672	0.5532
III	MTransQuest ⊗	0.7300	0.7226	0.6601	0.7402	0.8021	0.6289	0.7289	0.6091
	STransQuest ⊗	0.6901	0.6892	0.6451	0.7251	0.7649	0.5967	0.7041	0.5991
IV	MTransQuest ⊗ - Aug	**0.7399**	**0.7307**	**0.6792**	**0.7492**	**0.8109**	0.6367	**0.7398**	0.6156
	STransQuest ⊗ - Aug	0.7002	0.6901	0.6498	0.7301	0.7667	0.5991	0.7134	0.6098
V	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	0.3923
	Best system	0.6918	0.5923	0.6188	0.6819	0.7888	**0.6641**	0.7397	**0.7582**
VI	MTransQuest-B	0.6423	0.6354	0.5772	0.6531	0.7005	0.5483	0.6239	0.5002
	STransQuest-B	0.5987	0.5872	0.5012	0.5901	0.6572	0.5098	0.5762	0.4551

Table 9.1: Pearson correlation (ρ) between *TransQuest* algorithm predictions and human post-editing effort. The best result for each language by any method is highlighted in bold. Row **I** shows the results for XLM-R-large model and Row **II** shows the results for XLM-R-base. Row **III** and Row **IV** present the results for further fine-tuning strategies explained in Section 9.3. Row **V** shows the results of the baseline 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 **VI** presents the results of the multilingual BERT (mBERT) model in TransQuest Architectures.

MonoTransQuest gained ≈ 0.2-0.3 Pearson correlation boost over OpenKiwi in all the language pairs. Additionally, *MonoTransQuest* achieves ≈ 0.4 Pearson correlation boost over OpenKiwi in the low-resource language pair Ne-En. Similar to HTER, in DA *SiameseTransQuest* falls behind *MonoTransQuest* architecture, however still *SiameseTransQuest* gained ≈ 0.2-0.3 Pearson correlation boost over OpenKiwi in all the language pairs. This shows that simple STS models with cross-lingual embeddings outperforms the complex state-of-the-art QE models in predicting DA too.

		Low-resource		**Mid-resource**			**High-resource**	
	Method	Si-En	Ne-En	Et-En	Ro-En	Ru-En	En-De	En-Zh
I	MTransQuest	0.6525	0.7914	0.7748	0.8982	0.7734	0.4669	0.4779
	STransQuest	0.5957	0.7081	0.6804	0.8501	0.7126	0.3992	0.4067
II	MTransQuest-base	0.6412	0.7823	0.7651	0.8715	0.7593	0.4421	0.4593
	STransQuest-base	0.5773	0.6853	0.6692	0.8321	0.6962	0.3832	0.3975
III	MTransQuest ⊗	0.6661	0.8023	0.7876	0.8988	0.7854	0.4862	0.4853
	STransQuest ⊗	0.6001	0.7132	0.6901	0.8629	0.7248	0.4096	0.4159
IV	MTransQuest ⊗ - Aug	**0.6849**	**0.8222**	**0.8240**	**0.9082**	**0.8082**	**0.5539**	**0.5373**
	STransQuest ⊗ - Aug	0.6241	0.7354	0.7239	0.8621	0.7458	0.4457	0.4658
V	OpenKiwi	0.3737	0.3860	0.4770	0.6845	0.5479	0.1455	0.1902
VI	MTransQuest-B	NS	0.6452	0.6231	0.8351	0.6661	0.3765	0.3982
	STransQuest-B	NS	0.5368	0.5431	0.7652	0.5541	0.3356	0.3462

Table 9.2: Pearson correlation (ρ) between *TransQuest* algorithm predictions and human DA judgments. The best result for each language by any method is highlighted in bold. Row **I** shows the results for XLM-R-large model and Row **II** shows the results for XLM-R-base. Row **III** and Row **IV** present the results for further fine-tuning strategies explained in Section 9.3. Row **V** shows the results of the baseline method; OpenKiwi. Row **VI** presents the results of the multilingual BERT (mBERT) model in TransQuest Architectures.

If we consider the two architectures; *MonoTransQuest* and *SiameseTransQuest*, *MonoTransQuest* outperformed the *SiameseArchitecture* in all the language pairs in both aspects of sentence-level quality estimation. This is similar to the experiments we discussed for monolingual STS in Part I of the thesis. The two architectures have a trade-off between accuracy and efficiency. On an Nvidia Tesla K80 GPU, *MonoTransQuest* takes 4,480s on average to train on 7,000 instances, while *SiameseTransQuest* takes only 3,900s on average for the same experiment. On the same GPU, *MonoTransQuest* takes 35s on average to perform inference on 1,000 instances which takes *SiameseTransQuest* only 16s to do so. Therefore we recommend using *MonoTransQuest* where accuracy is valued over efficiency, and *SiameseTransQuest* where efficiency is prioritised above accuracy. Furthermore, as we discussed in Part I of the thesis, Siamese architecture outputs sentence vectors. Therefore finding the best quality translation in a collection of translations would take less time with the Siamese architecture. Since there is a growing interest[6] in the NLP community for energy efficient machine learning models, we decided to support both architectures in the TransQuest Framework.

With these results, we can answer our **RQ1:** Can the state-of-the-art STS models be applied in sentence-level QE tasks by changing the embeddings? We show that state-of-the-art STS methods based on cross-lingual transformer models outperform current state-of-the-art QE algorithms based on complex architectures like predictor-estimator in both aspects of sentence-level QE. We support this with the results for more than 15 language pairs with a wide variety from different domains and different MT types.

Additionally, Row VI in both Tables 9.1 and 9.2 shows the results of multilingual BERT (mBERT) in *MonoTransQuest* and *SiameseTransQuest* architectures. We used the same settings similar to XLM-R. The results show that XLM-R model outperforms the mBERT model in all the language pairs of both aspects in quality estimation. As shown in Row II in both Tables 9.1 and 9.2 even XLM-R-base model outperforms mBERT model in all the languages pairs

[6]Several workshops like *EMC*^{2} (Workshop on Energy Efficient Machine Learning and Cognitive Computing), SustaiNLP 2020 (Workshop on Simple and Efficient Natural Language Processing) has been organised on this aspect.

with both architectures. Therefore, we can safely assume that the cross-lingual nature of the XLM-R transformers had a clear impact on the results.

With this, we can answer our **RQ2:** Using cross-lingual embeddings with STS architecture proved advantageous rather than using multilingual embeddings. Therefore, state-of-the-art cross-lingual transformer models such as XLM-R should be utilised more in QE tasks.

9.3 Further Fine-tuning

In this section, we improve the results of TransQuest through two fine-tuning strategies; ensemble learning and data augmentation.

Ensemble Learning learning uses multiple learning algorithms to obtain better predictive performance than any of the constituent learning algorithms alone. We perform a weighted average ensemble for the output of the XLM-R-large and the output of the XLM-R-base for both architectures in TransQuest. We experimented on weights 0.8:0.2, 0.6:0.4, 0.5:0.5 on the output of XLM-R-large and output from XLM-R-base, respectively. Since the results from the XLM-R-base transformer model are slightly worse than those from the XLM-R-large, we did not consider the weight combinations that give higher weight to XLM-R-base transformer model results. We obtained the best results when we used the weights 0.8:0.2. We report the results from the two architectures from this step in Row III of Tables 9.1 and 9.2 as MTransQuest ⊗ and STransQuest ⊗.

As shown in Tables 9.1 and 9.2 both architectures in TransQuest with ensemble learning gained ≈ 0.01-0.02 Pearson correlation boost over the default settings for all the language pairs in both aspects of sentence-level QE.

Data Augmentation adds additional training instances for the training process. As we already experienced with STS experiments in Part I of the thesis, this often leads to better performance in the machine learning algorithms. Therefore, to experiment with how TransQuest performs with more data, we trained TransQuest on an augmented dataset. Alongside the training, development and testing datasets, the shared task organisers also provided the parallel sentences used to train the neural machine translation system in each language. In the data augmentation setting, we added the sentence pairs from that neural machine translation system training file to the training dataset we used to train TransQuest. In order to find the best setting for the data augmentation, we experimented with randomly adding 1000, 2000, 3000, up to 5000 sentence pairs. Since the ensemble learning performed better than XLM-R-large results of TransQuest, we conducted this data augmentation experiment on the ensemble learning setting. We assumed that the sentence pairs added from the neural machine translation system training file have maximum translation quality.

Up to 2000 sentence pairs, the results continued to get better. However, adding more than 2000 sentence pairs did not improve the results. We did not experiment with adding any further than 5000 sentence pairs to the training set since we were aware that adding more sentence pairs with the maximum translation quality to the training file will make it imbalanced and negatively affect the machine learning models' performance. We report the results from the two architectures from this step in Row IV of Tables 9.1 and 9.2 as MTransQuest $\otimes$-Aug and STransQuest $\otimes$-Aug.

Data augmentation provided the best results for all of the language pairs in both aspects of sentence-level QE. As shown in Tables 9.1 and 9.2, both architectures in TransQuest with the data augmentation setting gained $\approx$ 0.01-0.09 Pearson correlation boost over the XLM-R-large in all the language pairs. Additionally, for DA, *MTransQuest $\otimes$-Aug* achieves $\approx$ 0.09 Pearson correlation boost over TransQuest with XLM-R large in the high-resource language pair En-De, which is the biggest boost got for any language pair.

This answers our **RQ3:** *Can further fine-tuning strategies be used to improve the results?.* We show that ensemble learning and data augmentation can be used to improve the results. In fact, data augmentation combined with ensemble learning provided the best results for all the language pairs in both aspects of the sentence-level QE.

Finally, we submitted the best result we had with TransQuest (MonoTransQuest with ensemble learning and data augmentation) to WMT 2020 QE shared task 1 (Specia et al., 2020). The official results of the competition show that *TransQuest* won first place in all the language pairs in the *Sentence-Level Direct Assessment* task. *TransQuest* is the sole winner in En-Zh, Ne-En and Ru-En language pairs and the multilingual track. For the other language pairs (En-De, Ro-En, Et-En and Si-En), it shares first place with (Fomicheva et al., 2020a), whose results are not statistically different from ours. Therefore, TransQuest is the new state-of-the-art in sentence-level QE.

9.4 Error analysis

In an attempt to better understand the performance and limitations of *TransQuest* we carried out an error analysis on the results obtained on Romanian - English and Sinhala - English. The availability of native speakers determined the choice of language pairs we analysed to perform this analysis. We focused on cases where the difference between the predicted and expected scores was high. This included instances where the predicted score was underestimated and overestimated.

Analysis of the results does not reveal obvious patterns. The largest number of errors seem to be caused by named entities in the source sentences. In some cases, these entities are mishandled during the translation. The resulting sentences are usually syntactically correct but semantically odd. Typical examples are *RO: În urmă explorărilor Căpitanului James Cook, Australia și Noua Zeelandă au devenit ținte ale colonialismului britanic. (As a result of Captain James Cook's explorations, Australia and New Zealand have become the targets of British colonialism.)* - *EN: Captain James Cook, Australia and New Zealand have finally become the targets of British colonialism.* (expected: -1.2360, predicted: 0.2560) and *RO: O altă problemă importantă cu care trupele Antantei au fost obligate să se confrunte a fost malaria. (Another important problem that the Triple Entente troops had to face was malaria.)* - EN: *Another important problem that Antarctic troops* *had to face was malaria.* (expected: 0.2813, predicted: -0.9050). However, it is debatable whether the expected scores for these two pairs should be so different. Both of them have apparent problems and cannot be clearly understood without reading the source. For this reason, we would expect that both of them have low scores. Instances such as this also occur in the training data. As a result of this, *TransQuest* learns contradictory information, which in turn leads to errors at the testing stage.

A large number of problems are caused by incomplete source sentences or input sentences with noise. For example the pair *RO: thumbright250pxDrapelul cu fâșiile în poziție verticală (The flag with strips in upright position) - EN: ghtghtness 250pxDrapel with strips in upright position* has an expected score of 0.0595, but our method predicts -0.9786. Given that only *ghtghtness 250pxDrapel* is wrong in the translation, the predicted score is far too low. In an attempt to see how much this noise influences the result, we ran the system with the pair *RO: Drapelul cu fâșiile în poziție verticală - EN: Drapel with strips in upright position.* The prediction is 0.42132, which is more in line with our expectations, given that one of the words is not translated.

Similar to Ro-En, in Si-En, most problems seem to be caused by the presence of named entities in the source sentences. For example, in the English translation: *But the disguised Shiv will help them securely establish the statue.* (expected: 1.3618, predicted: -0.008), the correct English translation would be *But the disguised Shividru will help them securely establish the statue.*. Only the named entity *Shividru* is translated incorrectly. Therefore the annotators have annotated the translation with high quality. However, TransQuest fails to identify that. Similar scenarios can be found in English translations *Kamala Devi Chattopadhyay spoke at this meeting, Dr. Ann.* (expected:1.3177, predicted:-0.2999) and *The Warrior Falls are stone's, halting, heraldry and stonework rather than cottages. The cathedral manor is navigable places* (expected:0.1677, predicted:-0.7587). It is clear that the presence of the named entities seem to confuse the algorithm we used. Hence it needs to handle named entities in a proper way.

9.5 Conclusion

In this chapter, we introduced *TransQuest*, a new open-source framework for quality estimation based on cross-lingual transformers. *TransQuest* is implemented using PyTorch (Paszke et al., 2019) and HuggingFace (Wolf et al., 2020) and supports the training of sentence-level quality estimation systems on new data. It outperforms other open-source tools on both aspects of sentence-level quality estimation and yields new state-of-the-art quality estimation results.

We propose two architectures: *MonoTransQuest* and *SiameseTransQuest*. As we showed in Part I of the thesis, they provide state-of-the-art results for STS. However, neither of them have been previously explored in QE tasks. It is the first time that STS architectures have been proposed to the QE task, as far as we know. We further improve the results with data augmentation and ensemble learning. The best results we achieved in this chapter were submitted to WMT 2020 QE task 1, and it won first place in all the language pairs. We can conclude that TransQuest is state-of-the-art in QE now.

We have open-sourced the framework, and the pre-trained sentence-level QE models for 15 language pairs are freely available to the community in HuggingFace model hub. Upon releasing, TransQuest has caught wide attention from the community resulting in more than 9,000 downloads within the first year. Furthermore, TransQuest has been used as the baselines for many shared tasks, including the most recent edition of WMT QE shared task[7] and *Explainable Quality Estimation shared task* on the second workshop on Evaluation & Comparison of NLP Systems[8]. Not only among researchers, TransQuest is also popular in the industry. ModelFront, which is a leading company in translation risk prediction, lists TransQuest as an option on their website[9]. We believe that the simplicity of the architectures in TransQuest is the reason for this popularity.

Sentence-level QE models are challenging to interpret as they only provide a score for the whole translation without indicating where the errors are. To overcome this with TransQuest, we introduce minor changes to the sentence-level architecture to support word-level quality estimation in Chapter 10. One limitation with TransQuest is that the pre-trained QE models are big in size as they depend on transformer models. Therefore, managing several of them for each language pair would be chaotic in a real-world application. We seek solutions for that with multilingual QE models in Chapter 11.

[7]WMT 2021 QE shared task is available on http://statmt.org/wmt21/quality-estimation-task.html

[8]Explainable Quality Estimation shared task is available on https://eval4nlp.github.io/sharedtask.html

[9]ModelFront options are available on https://modelfront.com/options

As future work, we hope to explore cross-lingual embeddings more with TransQuest for the sentence-level QE task. For example, XeroAlign (Gritta and Iacobacci, 2021) provides a simple method for task-specific alignment of cross-lingual pre-trained transformers, which would be interesting to apply in QE tasks. Furthermore, Facebook has introduced large cross-lingual models recently; XLM-R XL and XLM-R XXL, which have improved results for many cross-lingual NLP tasks (Goyal et al., 2021). These models have the obvious disadvantage of being big in size, yet they would be interesting to apply in QE.

Chapter 10

Extending TransQuest for Word-Level QE

Translation quality can be estimated at different levels of granularity: word, sentence and document level (Ive et al., 2018). So far in this thesis, we have only explored sentence-level QE (Specia et al., 2020), in which QE models provide a score for each pair of source and target sentences. A more challenging task, which is currently receiving a lot of attention from the research community, is the word-level quality estimation which provides more fine-grained information about the quality of a translation, indicating which words from the source have been incorrectly translated in the target (good vs bad words), and whether the words inserted between these words are correct (good vs bad gaps). This information can be useful for post-editors by indicating the parts of a translation on which they have to focus more. Therefore, word-level QE solutions would certainly improve the efficiency of the post-editors.

Furthermore, as mentioned before, sentence-level QE models are difficult to explain as they only output a single score for the translation. The users face the difficulty of interpreting the score and how to make use of the information from the QE method. However, in contrast to that, word-level QE models provide more fine-grained information about the quality of a translation. As a result, they can improve the usability of the whole QE process. Since there is a growing interest in the NLP community for practical machine learning, we believe that having more usable QE models would improve the popularity of QE.

With these advantages, researchers have provided many solutions for word-level QE. Similar to sentence-level QE, state-of-the-art models in word-level QE are also relying on deep learning. As we explained in Chapter 8, most of the open-source quality estimation frameworks, such as OpenKiwi, uses the same architectures for both word-level and sentence-level QE. For example, the predictor-estimator architecture used in OpenKiwi provides the best results for word-level QE too. Therefore, word-level QE also suffers from the same limitation we mentioned in the last chapter. The architectures are very complex and need a lot of computing resources to train the models, which has seriously hindered the success of word-level QE in real-world applications (Ranasinghe et al., 2021b).

To overcome this, we propose a simple architecture to perform word-level QE. The proposed architecture is very similar to the *MonoTransQuest* architecture we introduced in the last chapter, which in turn was based on a state-of-the-art STS method. We only change the output layer of the *MonoTransQuest* architecture so that it can produce word-level qualities. This architecture is simple and effective when compared with the current state-of-the-art word-level QE architectures such as predictor-estimator. We believe that a simpler architecture can improve the popularity of word-level QE in real-world applications.

As mentioned in Chapter 8, word-level QE systems should have three features in them. *i.* Predict the qualities of the words in the target. *ii.* Predict the qualities of the words in the source. *iii.* Predict the qualities of the gaps in the target. As a result, WMT word-level QE shared task has three separate evaluation metrics focussing on each of the above features. However, most of the open-source word-level QE frameworks ignore predicting the quality of the gaps in the target sentence. For example, Marmot (Logacheva et al., 2016) only supports predicting the quality of the words in the target, and OpenKiwi (Kepler et al., 2019) only supports predicting the quality of the words in the source and the target. They completely ignore predicting the quality of the gaps. Despite that, predicting the quality of the gaps in the target would be important and useful for post-editors. Therefore, when we design the proposed architecture in this chapter, we considered all three features in word-level QE; predicting the quality of the words in the target, predicting the quality of the words in the source and predicting the quality of the gaps in the target. We believe an approach that supports every feature in word-level QE will be stronger than existing solutions.

We address three research questions in this chapter:

RQ1: Can existing state-of-the-art STS architecture be used to predict all the features in the word-level QE task by just modifying the embeddings and output layer?

RQ2: Do cross-lingual embeddings have an advantage over multilingual embeddings in the word-level QE task?

RQ3: Can the proposed model be further improved by performing ensemble learning?

The main contributions of this chapter are as follows.

1. We introduce a simple architecture to perform word-level quality estimation that predicts the quality of the words in the source sentence, target sentence and the gaps in the target sentence.

2. We evaluate it on eight different language pairs in which the word-level QE data was available, and we show that the proposed architecture outperforms the current state-of-the-art word-level QE frameworks such as Marmot (Logacheva et al., 2016) and OpenKiwi (Kepler et al., 2019).

3. We suggest further improvements to the model by performing ensemble learning.

4. We integrated the architecture with TransQuest, which already had two sentence-level architectures described in Chapter 9[1]. TransQuest framework was already popular with the NLP community and adding a word-level architecture to that boosted its value. Additionally the pre-trained word-level QE models for eight language pairs are freely available to the community[2].

The rest of this chapter is organised as follows. Section 10.1 discusses the methodology and the experiments conducted with eight language pairs in word-

[1]The public GitHub repository of TransQuest is available on `https://github.com/tharindudr/TransQuest`

[2]Pre-trained word-level QE models are available on `https://huggingface.co/models?filter=microtransquest`

level QE. Section 10.2 shows the results of three evaluation metrics used in word-level QE. Section 10.3 provides further fine-tuning strategies to improve the results. The chapter finishes with conclusions and ideas for future research directions in word-level QE.

10.1 Methodology

The proposed architecture for the word-level QE is very similar to the *MonoTransQuest* architecture in Chapter 9 where the source and the target are processed through a single transformer model. The difference here is since we need quality for each word, we do not use a pooling mechanism as we did with *MonoTransQuest*. Instead, we keep the state of the individual tokens to get their quality.

As we also need to consider the quality of the GAPs in word-level QE, we first add a new token to the tokeniser of the transformer called <GAP> which is inserted between the words in the target. We then concatenate the source and the target with a [SEP] token as in *MonoTransQuest* architecture and feed them into a single transformer. A simple linear layer is added on top of the word and <GAP> embeddings to predict whether it is "Good" or "Bad" as shown in Figure 10.1. Each of the softmax layers we used on top of the transformer model consists of two neurons. They provide two probabilities for each word or gap, stating the probability of being "Good" or "Bad". We consider the class with the maximum probability as the quality of a particular word or gap. As we integrated this architecture to the *TransQuest* framework and it provides quality for the

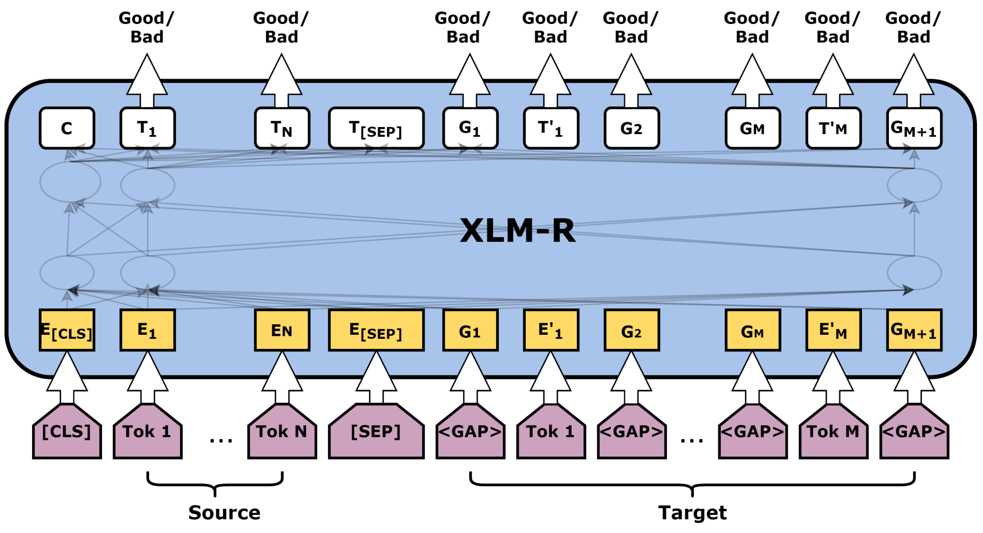


Figure 10.1: MicroTransQuest Architecture

smallest unit possible in QE, we named the proposed word-level architecture as *MicroTransQuest.*

Similar to the sentence-level QE architectures, we used cross-lingual transformer models for this architecture. As explained in Chapter 9, XLM-R provides state-of-the-art cross-lingual transformer models. There are two pre-trained XLM-R models released by HuggingFace's Transformers library (Wolf et al., 2020); XLM-R-base and XLM-R-large. We used both of these pre-trained models in our experiments[3]. Both of these pre-trained XLM-R models cover 104 languages (Conneau et al., 2020), making them potentially very useful to estimate the word-level translation quality for a large number of language pairs.

[3]XLM-R-large is available on https://huggingface.co/xlm-roberta-large and XLM-R-base is available on https://huggingface.co/xlm-roberta-base

10.1.1 Experimental Setup

We evaluated the architecture in word-level quality estimation using the data introduced in Chapter 8. We applied the same set of configurations for all the language pairs to ensure consistency between all the experiments. This also provides a good starting configuration for researchers who intend to use *MicroTransQuest* on a new language pair. We used a batch-size of eight, Adam optimiser with a learning rate $1e^{-5}$, and a linear learning rate warm-up over 10% of the training data. During the training process, the parameters of the XLM-R model and the subsequent layers were updated. The models were trained using only training data. Furthermore, they were evaluated while training once in every 300 training steps using an evaluation set with one-fifth of the training data instances. We performed early stopping if the evaluation loss did not improve over ten evaluation steps. All the models were trained for three epochs.

10.2 Results and Evaluation

For the evaluation, we used the approach proposed in the WMT shared tasks in which the classification performance is calculated using the multiplication of F1-scores ($F1_{MULTI}$) for the 'OK' and 'BAD' classes against the true labels independently: words in the target ('OK' for correct words, 'BAD' for incorrect words), gaps in the target ('OK' for genuine gaps, 'BAD' for gaps indicating missing words) and source words ('BAD' for words that lead to errors in the target, 'OK' for other words) (Specia et al., 2018) as explained in Chapter 8. In

WMT QE shared tasks, Word-level QE systems have been evaluated using all three of these evaluation metrics. Therefore, we follow the same process so that we can compare our results with the baselines and best systems of the respective shared task.

$F1_{MULTI}$ for the words in the target ($F1_{MULTI}$ $Target$), $F1_{MULTI}$ for the gaps in the target ($F1_{MULTI}$ $GAPS$) and $F1_{MULTI}$ for the words in the source ($F1_{MULTI}$ $Source$) are shown in Tables 10.1, 10.2 and 10.3 respectively. Before WMT 2019, organisers provided separate scores for gaps and words in the target, while after WMT 2019, they produce a single result for target gaps and words. Therefore, we report ($F1_{MULTI}$ $GAPS$) only on the language pairs released in WMT 2018 in Table 10.2. For the language pairs in WMT 2019 and 2020, we report a single result for the target gaps and words as ($F1_{MULTI}$ $Target$) in Table 10.1.

Raw I in Tables 10.1, 10.2 and 10.3 shows the results for *MicroTransQuest* with XLM-R-large and Raw II in Tables 10.1, 10.2 and 10.3 shows the results for *MicroTransQuest* with XLM-R-base. As can be seen in results, XLM-R-large always outperformed XLM-R-base. Therefore, we use the results we got from XLM-R-large for the following analysis.

In $F1_{MULTI}$ $Target$ evaluation metric in word-level QE, as shown in Table 10.1, *MicroTransQuest* outperforms OpenKiwi and Marmot in all the language pairs we experimented. Additionally, *MicroTransQuest* achieves ≈ 0.3 $F1_{MULTI}$ score boost over OpenKiwi in the En-Ru NMT. Furthermore, *MicroTransQuest* gained ≈ 0.15-0.2 $F1_{MULTI}$ score boost over Marmot in all the language pairs. Table 10.1 also

		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	MicroTransQuest	0.6081	0.5592	0.5939	0.5868	0.6485	0.5602	0.6348	0.5013
II	MicroTransQuest-base	0.5642	0.5132	0.5579	0.5431	0.6001	0.5202	0.5985	0.4698
III	MicroTransQuest ⊗	**0.6154**	**0.5672**	**0.6123**	**0.6003**	**0.6668**	0.5678	**0.6451**	0.5121
IV	Marmot	0.4449	NR	0.3445	0.4208	0.4373	NR	0.3653	NR
	OpenKiwi	NR	0.2412	NR	NR	NR	0.5583	NR	0.4111
	Best system	0.4449	0.4780	0.3618	0.4293	0.6012	**0.6415**	0.6246	**0.6186**
V	MicroTransQuest-B	0.5462	0.4987	0.5043	0.5132	0.5643	0.4892	0.5381	0.4371

Table 10.1: $F1_{MULTI}$ $Target$ between the algorithm predictions and human annotations. The best result for each language by any method is highlighted in bold. Row **I** shows the results for XLM-R-large model and Row **II** shows the results for XLM-R-base. Row **III** presents the results for further fine-tuning strategies explained in Section 10.3. Row **IV** shows the results of the baseline 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 *MicroTransQuest*.

gives the results of the best system submitted for a particular language pair. It is worth noting that the *MicroTransQuest* results surpass the best system in all the language pairs, with the exception of the En-De NMT and En-Zh NMT datasets.

In $F1_{MULTI}$ *GAPS* evaluation metric in word-level QE, as shown in Table 10.2 *MicroTransQuest* outperforms best systems submitted to the respective shared tasks in all the language pairs we experimented. *MicroTransQuest* achieves $\approx$ 0.05-0.2 $F1_{MULTI}$ score boost over best systems. Notably, *MicroTransQuest* achieves $\approx$ 0.2 $F1_{MULTI}$ score boost over the best system in En-De SMT. As we mentioned before, it should be noted that neither of the baselines, Marmot, nor OpenKiwi supports predicting the quality of *gaps* in the target sentence. Since *MicroTransQuest* supports this and produces state-of-the-results in this

		Mid-resource			High-resource	
	Method	En-Cs SMT	En-Lv SMT	En-Lv NMT	De-En SMT	En-De SMT
I	MicroTransQuest	0.2018	0.2356	0.1664	0.4203	0.4927
II	MicroTransQuest-base	0.1876	0.2132	0.1452	0.4098	0.4679
III	MicroTransQuest ⊗	**0.2145**	**0.2437**	**0.1764**	**0.4379**	**0.4982**
IV	Marmot	NS	NS	NS	NS	NS
	Best system	0.1671	0.1386	0.1598	0.3176	0.3161
V	MicroTransQuest-B	0.1778	0.2089	0.1387	0.3892	0.4371

Table 10.2: $F1_{MULTI}$ *GAPS* between the algorithm predictions and human annotations. The best result for each language by any method is highlighted in bold. Row **I** shows the results for XLM-R-large model and Row **II** shows the results for XLM-R-base. Row **III** presents the results for further fine-tuning strategies explained in Section 10.3. Row **IV** shows the results of the baseline methods and the best system submitted for the language pair in that competition. **NS** implies that a particular baseline does *not support* predicting quality of gaps. Row **V** presents the results of the multilingual BERT (mBERT) model in *MicroTransQuest*.

evaluation metric, we believe that using *MicroTransQuest* in word-level QE would be beneficial than using OpenKiwi and Marmot.

Finally, in $F1_{MULTI}$ *Source* evaluation metric, as shown in Table 10.3 *MicroTransQuest* outperforms OpenKiwi in all the language pairs we experimented. It should be noted that Marmot does not support predicting the quality of the words in the source. On the other hand, OpenKiwi supports this, but *MicroTransQuest* achieves ≈ 0.05-0.3 $F1_{MULTI}$ score boost in all the language pairs. Furthermore, *MicroTransQuest* results surpass the best system in all the language pairs with the exception of the En-De NMT and En-Zh NMT datasets.

With these results, we can answer our **RQ1:** *Can existing state-of-the-art*

		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	MicroTransQuest	0.5327	0.5543	0.4945	0.4880	0.4824	0.4040	0.5269	0.4456
II	MicroTransQuest-base	0.5134	0.5287	0.4652	0.4571	0.4509	0.3876	0.5012	0.4185
III	MicroTransQuest ⊗	**0.5431**	**0.5640**	**0.5076**	**0.4892**	**0.4965**	0.4145	**0.5387**	0.4578
IV	Marmot	NS	NR	NS	NS	NS	NR	NS	NR
	OpenKiwi	NR	0.2647	NR	NR	NR	0.3729	NR	0.3717
	Best system	0.3937	0.4541	0.4945	0.3614	0.3200	**0.4462**	0.3368	**0.5672**
V	MicroTransQuest-B	0.4987	0.5098	0.4441	0.4256	0.4187	0.3567	0.4672	0.3985

Table 10.3: $F1_{MULTI}$ *Source* between the algorithm predictions and human annotations. The best result for each language by any method is highlighted in bold. Row **I** shows the results for XLM-R-large model and Row **II** shows the results for XLM-R-base. Row **III** presents the results for further fine-tuning strategies explained in Section 10.3. Row **IV** shows the results of the baseline 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 *MicroTransQuest.*

STS architecture be used to predict all features in the word-level QE task by just modifying the embeddings and the output layer? We show that the state-of-the-art STS method based on cross-lingual transformer models can be used in word-level QE by changing the output layer. It outperforms current state-of-the-art word-level QE methods such as OpenKiwi that relies on complex neural network architectures. Our proposed architecture is not only simple, but also achieves state-of-the-art results in all the language pairs we experimented. Furthermore, *MicroTransQuest* is the only architecture that supports predicting the quality of the gaps in target. We believe that the way we designed the architecture based on state-of-the-art STS architectures gave us the ability to support predicting the quality of the gaps in target with *MicroTransQuest.*

Row VI in Tables 10.1, 10.2 and 10.3 shows the results of multilingual BERT (mBERT) in *MicroTransQuest* architecture. We used the same settings similar to XLM-R. The results show that the XLM-R model outperforms the mBERT model in all the language pairs. As shown in Row II in Tables 10.1, 10.2 and 10.3 even XLM-R-base model outperform mBERT model in all the languages pairs. Therefore, we can safely assume that the cross-lingual nature of the XLM-R transformers had a clear impact on the results. This observation is similar to what we experienced in Chapter 9 with sentence-level QE experiments.

With this, we can answer our **RQ2:** Using cross-lingual embeddings with STS architecture proved advantageous rather than using multilingual embeddings. As far as we know, this is the first time XLM-R was used in a task related to detecting missing words. With the results we got from predicting the quality of the gaps in target, we show that XLM-R can be used in tasks similar to detecting missing words.

10.3 Further Improvements

In this section, we improve the results of *MicroTransQuest* through ensemble learning as we did with sentence-level quality estimation. As mentioned before in Section 10.1 softmax layer provides two probabilities for each word or gap. We calculated these probabilities for each word using XLM-R-large and XLM-R base. Then we performed a weighted average ensemble on these probabilities, and we consider the class with the highest probability after performing the ensemble as the quality of the word or gap. We experimented on weights

0.8:0.2, 0.6:0.4, 0.5:0.5 on the output of XLM-R-large and output from XLM-R-base, respectively. Since the results we got from XLM-R-base transformer model are slightly worse than the results we got from XLM-R-large we did not consider the weight combinations that gives higher weight to XLM-R-base transformer model results. We obtained the best results when we used the weights 0.6:0.4. We report the results from this step in Row III of Tables 10.1, 10.2 and 10.3 as MicroTransQuest ⊗.

As shown in Tables 10.1, 10.2 and 10.3 ensemble learning improved the results for *MicroTransQuest* in all three evaluation metrics. For all the language pairs $F1_{MULTI}$ $Target$, $F1_{MULTI}$ *GAPS*, and $F1_{MULTI}$ *Source* scores gained ≈ 0.01-0.02 boost with ensemble learning. This answers our **RQ3:** *Can the proposed model be further improved by performing ensemble learning?* We show that ensemble learning can be used to improve the word-level QE results similar to sentence-level QE. In fact, ensemble learning provided the best results for *MicroTransQuest* in all the language pairs we experimented.

10.4 Conclusion

In this chapter, we explored word-level QE with transformers. We introduced a new architecture based on transformers to perform word-level QE. The proposed architecture is very similar to the sentence-level QE architecture; *MonoTransQuest*. However, the output layers are different to *MonoTransQuest* so that they keep the state of the individual tokens to get their quality. We evaluated the proposed architecture; *MicroTransQuest* on eight language pairs where the word-level QE data was available. It outperforms other open-source tools like OpenKiwi and Marmot on all three evaluations metrics of word-level quality estimation and yields new state-of-the-art word-level QE results. Furthermore, *MicroTransQuest* is the only open-source architecture that supports predicting the quality of the gaps in the target sentence. The architecture we proposed in this chapter is very simple compared to the predictor-estimator architecture in OpenKiwi, yet this architecture produces better results. We further improved the results with ensemble learning showed that *MicroTransQuest* outperforms the majority of the best systems submitted for that language in each shared task. We can conclude that state-of-the-art STS architecture can also be used in word-level QE by doing small modifications to the output layers.

The implementation of the architecture, which is based on PyTorch (Paszke et al., 2019) and Hugging Face (Wolf et al., 2020), has been integrated into the TransQuest framework (Ranasinghe et al., 2020c) which won the WMT 2020 QE task (Specia et al., 2020) on sentence-level direct assessment (Ranasinghe et al., 2020b)[4]. The pre-trained word-level QE models for eight language pairs are available to the public on HuggingFace model hub.

One limitation of the proposed architecture is that it only predicts word-level qualities. Managing two models to predict the word-level and sentence-level qualities separately would be chaotic in some situations as the two models can produce contradictory predictions. Therefore, in the future, we hope to explore

[4]The details about the word-level QE architecture in TransQuest is available on http://tharindu.co.uk/TransQuest/architectures/word_level_architecture/

multi-task learning for word-level and sentence-level QE (Caruana, 1997). The two tasks are related as word-level quality contributes to the sentence-level quality in general. Therefore we believe that a multi-task architecture that can learn both tasks simultaneously can be advantageous.

As discussed at multiple points in this thesis, pre-trained models based on transformers are big in size. Therefore, managing several of them for each language pair would be chaotic in a real-world application for word-level QE. As a solution for that, we explore multilingual word-level QE models in Chapter 11 in Part III of the thesis.

Chapter 11

Multilingual Quality Estimation with TransQuest

Machine translation quality estimation is traditionally framed as a supervised machine learning problem (Kepler et al., 2019; Lee, 2020) 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 having machine learning models for each language pair separately.

This traditional approach has obvious drawbacks. As mentioned before, this process requires training data for each language pair. However, the training data publicly available to build QE models is limited to very few language pairs, making 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 (Ranasinghe et al., 2020c).

To understand this scale, consider a real-world application where it is required to build a quality estimation solution for the European Parliament. The

European Parliament has 24 languages, resulting in 24*23 language pairs equal to 552 language pairs. A traditional bilingual 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 multilingual learning has been applied to many tasks (Ranasinghe and Zampieri, 2020; Ranasinghe and Zampieri, 2021) including NMT (Nguyen and Chiang, 2017; Aharoni et al., 2019), multilingual approaches have been rarely used in QE (Sun et al., 2020a). Therefore, in this chapter we explore multilingual models with the *TransQuest* architectures we introduced in Chapters 9 and 10 for sentence-level QE and word-level QE respectively. Since we used a cross-lingual transformer model that supports 104 languages (Conneau et al., 2020) it is possible to explore multilingual learning with the same setup.

Usually, the neural machine learning models are *hungry for data.* They need a lot of annotated data for the training process, which is a challenge for the low resource languages. Recently, researchers are trying to exploit this behaviour with learning paradigms such as few-shot and zero-shot learning. What we define as few-shot learning in this chapter is the process where the QE model only sees a few instances from a certain language pair in the training process (Wang et al., 2020b) while in zero-shot learning, QE model would not see any instances from a certain language pair (Chang et al., 2008) in the training process. Even though few-shot and zero-shot learning has been popular in machine learning applications, including NLP, they have not been explored with QE. Exploring them would benefit the low resource languages, which the QE training data is difficult to find. Therefore, this chapter inspects how the bilingual and multilingual QE models behave in few-shot and zero-shot learning environments.

As far as we know, this is the first study done on multilingual word-level and sentence-level QE. We address three research questions in this chapter:

RQ1: Do multilingual models based on existing state-of-the-art sentence-level, and word-level QE architectures perform competitively with the related bilingual models?

RQ2: How do the bilingual and multilingual models perform in a zero-shot environment, and what is the impact of source-target direction, domain and MT type for zero-shot learning?

RQ3: Do multilingual QE models perform better with a limited number of training instances (Few-shot learning) for an unseen language pair?

The main contributions of this Chapter are,

1. We explore multilingual, sentence-level and word-level quality estimation with the proposed architectures in *TransQuest*. We show that multilingual models are competitive with bilingual models.

2. We inspect few-shot and zero-shot sentence-level and word-level quality estimation with the bilingual and multilingual models. We report how the source-target direction, domain and MT type affect the predictions for a new language pair.

3. The code and the pre-trained multilingual models of *TransQuest* are publicly available to the community[1].

The rest of this chapter is organised as follows. Section 11.1 discusses the methodology and the multilingual experiments done with 15 language pairs in sentence-level QE and word-level QE. Section 11.2 shows the results and Sections 11.2.2 and 11.2.3 provide further analysis on zero-shot and few-shot learning. The chapter finishes with conclusions and ideas for future research directions in multilingual QE.

11.1 Methodology

As mentioned before, we conducted the experiments with the architectures explored in Chapters 9 and 10. For the multilingual sentence-level experiments, we used the *MonoTransQuest* architecture introduced in Chapter 9 which outperformed other open-source QE frameworks and best systems submitted to the shared tasks in the majority of the language pairs. We experimented with both aspects of sentence-level QE, HTER and DA with all the datasets introduced in Chapter 8. For the word-level experiments, we used the *MicroTransQuest*

[1]The pre-trained multilingual QE models are available on the HuggingFace model repository on `https://huggingface.co/TransQuest`

architecture which again outperformed other open-source QE frameworks and best systems submitted to the shared tasks in the majority of the language pairs in word-level QE. As the word-level QE data, we considered all the word-level QE datasets introduced in Chapter 8.

For the experiments, we considered XLM-R large model and did not use the ensemble models to keep the multilingual experiments simpler. We applied the same set of configurations for all the training processes to ensure consistency between the experiments. We used a batch size of eight, Adam optimiser and a linear learning rate warm-up over 10% of the training data. During the training process, the parameters of the XLM-R-large model, as well as the parameters of the subsequent layers, were updated. The models were trained using only training data. Furthermore, they were evaluated while training once in every 100 training steps using an evaluation set with one-fifth of the training data rows. We performed early stopping if the evaluation loss did not improve over ten evaluation steps. All the models were trained for three epochs. We used a learning rate of $2e^{-5}$ for the sentence-level experiments while for word-level experiments it was $1e^{-5}$.

11.2 Results

In the following sections, we explore different multilingual QE settings and compare the multilingual results to the results we got with supervised, bilingual QE models.

For the sentence-level evaluations, we used Pearson correlation, the same evaluation metric we used for bilingual sentence-level QE in Chapter 9, which was used for WMT QE shared tasks. Results for multilingual sentence-level QE experiments with HTER and DA are shown in Tables 11.1 and 11.2. For the word-level evaluations, we used the same evaluation metrics we used for bilingual word-level QE in Chapter 10 which in turn was used for the WMT QE shared tasks. Results for multilingual word-level QE experiments with regards to $F1_{MULTI}$ for words in target ($F1_{MULTI}$ $Target$), $F1_{MULTI}$ for gaps in the target ($F1_{MULTI}$ $GAPS$) and $F1_{MULTI}$ for words in source ($F1_{MULTI}$ $Source$) are shown in Tables 11.3, 11.4 and 11.5 respectively.

The values displayed diagonally across the Row I of Tables 11.1, 11.2, 11.3, 11.4 and 11.5 show the results for supervised, bilingual, QE models where the model was trained on the training set of a particular language pair and tested on the test set of the same language pair. This is the exact result we reported in the Row I of Tables 9.1, 9.2, 10.1, 10.2 and 10.3 in Chapters 9 and 10, which we got with *MonoTransQuest* and *MicroTransQuest* using XLM-R-large model (Conneau et al., 2020) for sentence-level and word-level.

11.2.1 Multilingual QE

First, We combined instances from the training sets of all the language pairs where the sentence-level HTER data was available and built a single QE model with *MonoTransQuest*. We evaluated this multilingual model on the test sets of all the language pairs. We repeat the same for sentence-level DA QE with *MonoTransQuest* and word-level QE with *MicroTransQuest*. Our results, displayed

		IT			Pharmaceutical			Wiki	
	Train Language(s)	En-Cs SMT	En-De SMT	En-Ru NMT	De-En SMT	En-LV NMT	En-Lv SMT	En-De NMT	En-Zh NMT
I	En-Cs SMT	**0.7207**	(-0.06)	(-0.07)	(-0.13)	(-0.02)	(-0.01)	(-0.11)	(-0.10)
	En-De SMT	(-0.01)	0.7137	(-0.04)	(-0.12)	(-0.04)	(-0.05)	(-0.07)	(-0.07)
	En-Ru NMT	(-0.12)	(-0.15)	**0.7126**	(-0.13)	(-0.01)	(-0.02)	(-0.08)	(-0.07)
	De-En SMT	(-0.39)	(-0.29)	(-0.34)	**0.7939**	(-0.27)	(-0.31)	(-0.26)	(-0.27)
	En-LV NMT	(-0.11)	(-0.13)	(-0.02)	(-0.11)	0.7394	(-0.01)	(0.08)	(-0.07)
	En-Lv SMT	(-0.03)	(-0.09)	(-0.08)	(-0.15)	(-0.01)	0.6592	(-0.13)	(-0.13)
	En-De NMT	(-0.11)	(-0.07)	(-0.02)	(-0.12)	(-0.01)	(-0.02)	0.5994	(-0.04)
	En-Zh NMT	(-0.21)	(-0.18)	(-0.02)	(-0.18)	(-0.02)	(-0.07)	(-0.08)	0.6119
II	All	0.7111	**0.7300**	0.7012	0.7878	**0.7450**	**0.7141**	0.5982	0.6092
	All-1	(-0.01)	(-0.04)	(-0.02)	(-0.11)	(-0.01)	(-0.01)	(-0.01)	(-0.03)
III	Domain	0.7001	0.7256	0.6987	0.7754	0.7412	0.7065	0.5764	0.5671
IV	SMT/NMT	0.6998	0.7143	0.6998	0.7642	0.7319	0.6872	0.5671	0.5601
V	Quest++	0.3943	0.3653	NR	0.3323	0.4435	0.3528	NR	NR
	OpenKiwi	NR	NR	0.5923	NR	NR	NR	0.3923	0.5058
	Best system	0.6918	0.7397	0.5923	0.7888	0.6819	0.6188	**0.7582**	**0.6641**

Table 11.1: Pearson correlation (ρ) between the *MonoTransQuest* predictions and human post-editing effort in multilingual experiments. The best result for each language by any method is highlighted in bold. Rows **I**, **II**, **III** and **IV** indicate the different multilingual settings. Row **V** shows the results of the baselines 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 coloured in grey and the value shows the difference between the best result in that Row for that language pair and itself.

in Row II ("All") of Tables 11.1, 11.2, 11.3, 11.4 and 11.5 show that multilingual models perform on par with the bilingual models or even better for some language pairs in all the evaluation metrics with both sentence-level and word-level. For example, in sentence-level HTER experiments as shown in Table 11.1 multilingual sentence-level QE model outperforms the bilingual sentence-level QE model in three language pairs; En-De SMT, En-Lv SMT and En-Lv NMT. In word-level also, as shown in Table 11.3, multilingual word-level QE model outperforms the bilingual word-level QE model in all the language pairs except

		Low-resource		**Mid-resource**			**High-resource**	
	Train Language(s)	Si-En	Ne-En	Et-En	Ro-En	Ru-En	En-De	En-Zh
I	Si-En	0.6525	(-0.05)	(-0.08)	(-0.15)	(-0.07)	(-0.13)	(-0.13)
	Ne-En	(-0.10)	0.7914	(-0.06)	(-0.08)	(-0.08)	(-0.10)	(-0.11)
	Et-En	(-0.07)	(-0.10)	0.7748	(-0.20)	(-0.08)	(-0.10)	(-0.08)
	Ro-En	(-0.02)	(-0.04)	(-0.02)	**0.8982**	(-0.08)	(-0.10)	(-0.14)
	Ru-En	(-0.11)	(-0.16)	(-0.19)	(-0.26)	0.7734	(-0.04)	(-0.09)
	En-De	(-0.32)	(-0.51)	(-0.39)	(-0.51)	(-0.35)	**0.4669**	(-0.17)
	En-Zh	(-0.16)	(-0.24)	(-0.19)	(-0.36)	(-0.17)	(-0.02)	**0.4779**
II	All	0.6526	0.7581	0.7574	0.8856	0.7521	0.4420	0.4646
	All-1	(-0.02)	(-0.02)	(-0.02)	(-0.03)	(-0.02)	(-0.02)	(-0.05)
III	OpenKiwi	0.3737	0.3860	0.4770	0.6845	0.5479	0.1455	0.1902

Table 11.2: Pearson correlation (ρ) between the *MonoTransQuest* predictions and human DA judgments in multilingual experiments. The best result for each language by any method is highlighted in bold. Rows **I** and **II** indicate the different multilingual settings. Row **III** shows the results of the baselines and the best system submitted for the language pair in that competition. Zero-shot results are coloured in grey and the value shows the difference between the best result in that Row for that language pair and itself.

En-Zh NMT, En-Ru NMT and De-En SMT with regard to Target F1-Multi. Similar observations can be made in other word-level evaluation metrics in Tables 11.4 and 11.5.

We also investigate whether combining language pairs that share either the same domain or MT type can be more beneficial since it is possible that the learning process is better when language pairs share certain characteristics. We could only conduct this experiment in sentence-level HTER QE and word-level QE as sentence-level DA QE datasets are from the same domain and MT type. However, as shown in Row III and IV of Tables 11.1, 11.3, 11.4 and 11.5, for

		IT			**Pharmaceutical**			**Wiki**	
	Train Language(s)	En-Cs SMT	En-De SMT	En-Ru NMT	De-En SMT	En-LV NMT	En-Lv SMT	En-De NMT	En-Zh NMT
I	En-Cs SMT	0.6081	(-0.07)	(-0.09)	(-0.15)	(-0.02)	(-0.01)	(-0.10)	(-0.11)
	En-De SMT	(-0.01)	0.6348	(-0.07)	(-0.14)	(-0.06)	(-0.04)	(-0.06)	(-0.09)
	En-Ru NMT	(-0.14)	(-0.16)	**0.5592**	(-0.12)	(-0.01)	(-0.03)	(-0.09)	(-0.08)
	De-En SMT	(-0.43)	(-0.33)	(-0.31)	**0.6485**	(-0.29)	(-0.32)	(-0.25)	(-0.28)
	En-LV NMT	(-0.12)	(-0.14)	(-0.03)	(-0.12)	0.5868	(-0.01)	(0.09)	(-0.08)
	En-Lv SMT	(-0.04)	(-0.10)	(-0.09)	(-0.16)	(-0.01)	0.5939	(-0.15)	(-0.14)
	En-De NMT	(-0.11)	(-0.08)	(-0.02)	(-0.14)	(-0.02)	(-0.04)	0.5013	(-0.06)
	En-Zh NMT	(-0.19)	(-0.17)	(-0.03)	(-0.16)	(-0.03)	(-0.06)	(-0.07)	0.5402
II	All	**0.6112**	**0.6583**	0.5558	0.6221	**0.5991**	**0.5980**	0.5101	0.5229
	All-1	(-0.01)	(-0.05)	(-0.02)	(-0.12)	(-0.01)	(-0.01)	(-0.01)	(-0.05)
III	Domain	0.6095	0.6421	0.5560	0.6331	0.5892	0.5951	0.5021	0.5210
IV	SMT/NMT	0.6092	0.6410	0.5421	0.6320	0.5885	0.5934	0.5010	0.5205
V	Marmot	0.4449	0.3630	NR	0.4373	0.4208	0.3445	NR	NR
	OpenKiwi	NR	NR	0.2412	NR	NR	NR	0.4111	0.5583
	Best system	0.4449	0.6246	0.4780	0.6012	0.4293	0.3618	**0.6186**	**0.6415**

Table 11.3: $F1_{MULTI}$ $Target$ between the *MicroTransQuest* predictions and human annotations in multilingual experiments. The best result for each language by any method is highlighted in bold. Row **I**, **II**, **III** and **IV** indicate the different multilingual settings. Row **V** shows the results of the baselines 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 coloured in grey and the value shows the difference between the best result in that Row for that language pair and itself.

the majority of the language pairs, specialised multilingual QE models built on certain domains or MT types do not perform better than multilingual models which contain all the data.

With these observations, we answer our **RQ1:** Multilingual models based on existing state-of-the-art QE architectures perform competitively with the related bilingual models, and in some of the language pairs, multilingual models even outperformed the related bilingual models.

		IT		Pharmaceutical		
	Train Language(s)	En-Cs SMT	En-De SMT	De-En SMT	En-LV NMT	En-Lv SMT
I	En-Cs SMT	0.2018	(-0.08)	(-0.15)	(-0.02)	(-0.01)
	En-De SMT	(-0.08)	0.4927	(-0.14)	(-0.06)	(-0.04)
	En-Ru NMT	(-0.14)	(-0.15)	(-0.12)	(-0.01)	(-0.03)
	De-En SMT	(-0.18)	(-0.33)	**0.4203**	(-0.29)	(-0.32)
	En-LV NMT	(-0.16)	(-0.15)	(-0.12)	0.1664	(-0.01)
	En-Lv SMT	(-0.11)	(-0.11)	(-0.16)	(-0.01)	0.2356
	En-De NMT	(-0.17)	(-0.09)	(-0.14)	(-0.02)	(-0.04)
	En-Zh NMT	(-0.15)	(-0.16)	(-0.16)	(-0.03)	(-0.06)
II	All	**0.2118**	**0.5028**	0.4189	**0.1772**	**0.2388**
	All-1	(-0.03)	(-0.08)	(-0.14)	(-0.01)	(-0.01)
III	Domain	0.2112	0.4951	0.4132	0.1685	0.2370
IV	SMT/NMT	0.2110	0.4921	0.4026	0.1671	0.2289
V	Marmot	NS	NS	NS	NS	NS
	Best system	0.1671	0.3161	0.3176	0.1598	0.1386

Table 11.4: $F1_{MULTI}$ *GAPS* between *MicroTransQuest* predictions and human annotations in multilingual experiments. The best result for each language by any method is highlighted in bold. Row **I**, **II**, **III** and **IV** indicate the different multilingual settings. Row **V** shows the results of the baselines and the best system submitted for the language pair in that competition. **NS** implies that a particular result was *not supported* by the respective baseline. Zero-shot results are coloured in grey and the value shows the difference between the best result in that Row for that language pair and itself.

11.2.2 Zero-shot QE

We performed zero-shot quality estimation to test whether a QE model trained on a particular language pair can be generalised to other language pairs, different domains and MT types. We used the QE model trained on a particular language pair and evaluated it on the test sets of the other language pairs. Non-diagonal values of Row I in Tables 11.1, 11.2, 11.3, 11.4 and 11.5 show how each QE model

		IT			Pharmaceutical			Wiki	
	Train Language(s)	En-Cs SMT	En-De SMT	En-Ru NMT	De-En SMT	En-LV NMT	En-Lv SMT	En-De NMT	En-Zh NMT
I	En-Cs SMT	0.5327	(-0.07)	(-0.09)	(-0.17)	(-0.02)	(-0.01)	(-0.12)	(-0.13)
	En-De SMT	(-0.01)	0.5269	(-0.08)	(-0.14)	(-0.06)	(-0.05)	(-0.08)	(-0.09)
	En-Ru NMT	(-0.14)	(-0.18)	**0.5543**	(-0.14)	(-0.01)	(-0.03)	(-0.09)	(-0.08)
	De-En SMT	(-0.42)	(-0.33)	(-0.31)	**0.4824**	(-0.29)	(-0.32)	(-0.23)	(-0.28)
	En-LV NMT	(-0.12)	(-0.14)	(-0.03)	(-0.12)	0.4880	(-0.01)	(0.09)	(-0.08)
	En-Lv SMT	(-0.04)	(-0.11)	(-0.09)	(-0.17)	(-0.02)	0.4945	(-0.15)	(-0.14)
	En-De NMT	(-0.11)	(-0.08)	(-0.02)	(-0.15)	(-0.03)	(-0.04)	0.4456	(-0.06)
	En-Zh NMT	(-0.19)	(-0.17)	(-0.03)	(-0.18)	(-0.05)	(-0.06)	(-0.07)	0.4040
II	All	**0.5442**	**0.5445**	0.5535	0.4791	**0.4983**	**0.5005**	0.4483	0.4053
	All-1	(-0.02)	(-0.06)	(-0.03)	(-0.16)	(-0.01)	(-0.01)	(-0.01)	(-0.04)
III	Domain	0.5421	0.5421	0.5259	0.4672	0.4907	0.4991	0.4364	0.4021
IV	SMT/NMT	0.5412	0.5412	0.5230	0.4670	0.4889	0.4932	0.4302	0.4012
V	Marmot	NS	NS	NR	NS	NS	NS	NR	NR
	OpenKiwi	NR	NR	0.2647	NR	NR	NR	0.3717	0.3729
	Best system	0.3937	0.3368	0.4541	0.3200	0.3614	0.4945	**0.5672**	**0.4462**

Table 11.5: $F1_{MULTI}$ *Source* between *MicroTransQuest* predictions and human annotations in multilingual experiments. The best result for each language by any method is highlighted in bold. Row **I**, **II**, **III** and **IV** indicate the different multilingual settings. Row **V** shows the results of the baselines and the best system submitted for the language pair in that competition. **NR** implies that a particular result was *not resported* by the organisers and **NS** implies that a particular result was *not supported* by the respective baseline. Zero-shot results are coloured in grey and the value shows the difference between the best result in that row for that language pair and itself.

performed on other language pairs. For better visualisation, the non-diagonal values of Row I in Tables 11.1, 11.2, 11.3, 11.4 and 11.5 show by how much the score changes when the zero-shot QE model is used instead of the bilingual QE model. As can be seen, the scores decrease, but this decrease is negligible and is to be expected. For most pairs, the QE model that did not see any training instances of that particular language pair outperforms the baselines that were trained extensively on that particular language pair. Further analysis of the results shows that zero-shot QE performs better when the language pair shares some properties such as domain, MT type or language direction. For example, in word-level QE, En-De SMT $\Rightarrow$ En-Cs SMT is better than En-De NMT $\Rightarrow$ En-Cs SMT and En-De SMT $\Rightarrow$ En-De NMT is better than En-Cs SMT $\Rightarrow$ En-De NMT in $F1_{MULTI}\ Target$. Similar observations can be made on other evaluation metrics too.

We also experimented with zero-shot QE with multilingual QE models. For sentence-level HTER QE, sentence-level DA QE and word-level QE separately, we trained a multilingual model in all the language pairs except one and performed prediction on the test set of the language pair left out. In Row II ("All-1") of Tables 11.1, 11.2, 11.3, 11.4 and 11.5, we show its difference to the multilingual QE model. This also provides competitive results for the majority of the languages, proving it is possible to train a single multilingual QE model and extend it to a multitude of languages and domains. This approach provides better results than performing transfer learning from a bilingual model.

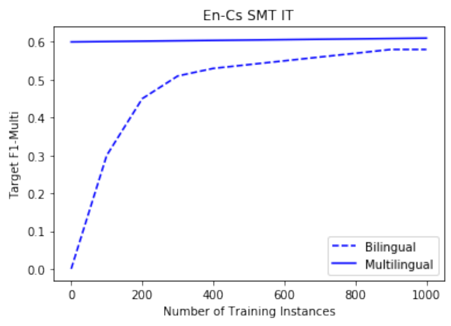
One limitation of the zero-shot QE is its inability to perform when the language direction changes. In the scenario where we performed zero-shot learning from De-En SMT to other language pairs in sentence-level HTER QE and word-level QE, results degraded considerably from the bilingual result. Similarly, the performance is rather poor when we test De-En for the multilingual zero-shot experiment as the direction of all the other pairs used for training differs. These observations are similar in sentence-level DA experiments with En-De and En-Zh.

With these observations, we answer our **RQ2:** zero-shot QE with state-of-the-art QE models provide very competitive results to language pairs which they did not see in the training process. Furthermore, multilingual models provide better zero-shot results than bilingual models.

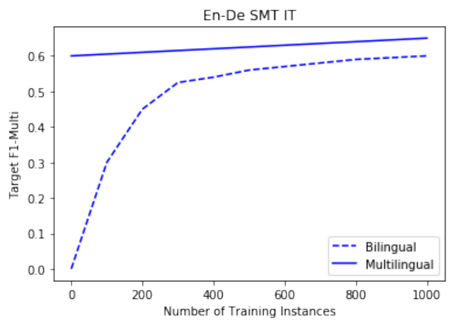
11.2.3 Few-shot QE

We also evaluated how the QE models behave with a limited number of training instances. For each language pair, we initiated the weights of the bilingual model with those of the relevant All-1 QE and trained it on 100, 200, 300 and up to 1000 training instances. We compared the results with those obtained, having trained the QE model from scratch for that language pair. The results in Figure 11.1 show that All-1 or the multilingual model performs well above the QE model trained from scratch (Bilingual) when there is a limited number of training instances available. Even for the De-En language pair in sentence-level HTER QE and word-level QE, for which we had comparatively poor zero-shot results, the multilingual model provided better results with a few training instances. It seems that having the model weights already fine-tuned in the multilingual model provides an additional boost to the training process, which is advantageous in a few-shot scenario.

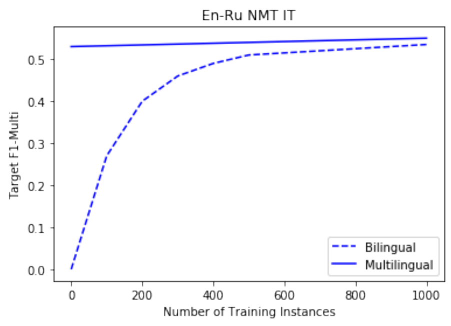
With these findings, we answer our **RQ3:** multilingual QE models perform better with a limited number of training instances (Few-shot learning) for an unseen language pair in both sentence-level and word-level QE. It is always better to transfer the weights from a multilingual QE model than to train the



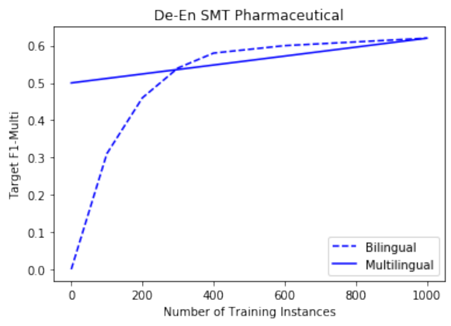
(a) En-Cs SMT IT



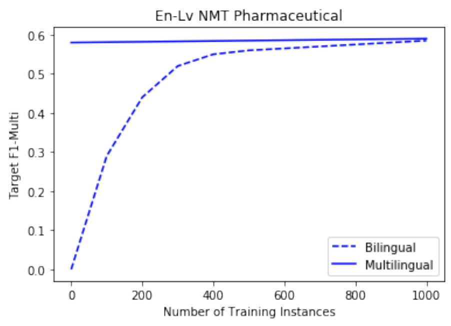
(b) En-De SMT IT



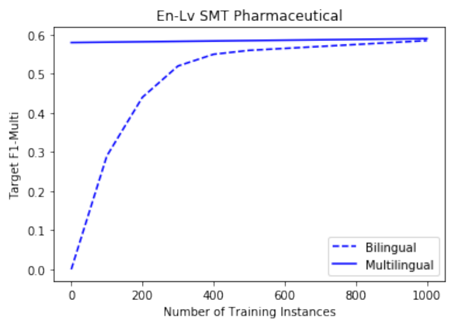
(c) En-Ru NMT IT



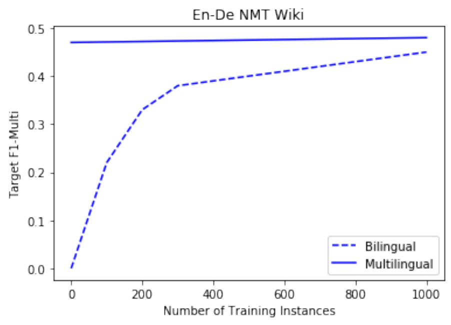
(d) De-En SMT Pharmaceutical



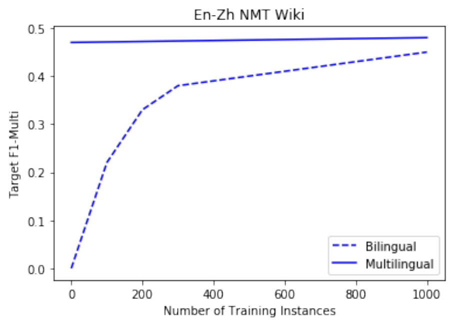
(e) En-Lv NMT Pharmaceutical



(f) En-Lv SMT Pharmaceutical



(g) En-De NMT Wiki



(h) En-Zh NMT Wiki

Figure 11.1: Few-shot learning Results for Word-Level QE. We report $F1_{MULTI}$ *Target* scores against Number of training instances for multilingual and bilingual models.

weights from scratch for a new language pair.

11.3 Conclusion

The traditional way of having a single QE model for each language pair has many limitations; *i.* They need to have annotated training data for each language pair which can be costly, *ii.* Managing several QE models at the same time can be chaotic. These limitations can hinder the ability of the state-of-the-art QE models to be applied in real-world applications. As a solution to that, we explored multilingual QE with state-of-the-art QE models. We used the sentence-level and word-level QE architectures in *TransQuest* and evaluated them in different multilingual settings.

In our experiments, we observed that multilingual QE models deliver excellent results on the language pairs they were trained on. In addition, the multilingual QE models perform well in the majority of the zero-shot scenarios where the multilingual QE model is tested on an unseen language pair. Furthermore, multilingual models perform very well with few-shot learning on an unseen language pair compared to training from scratch for that language pair, proving that multilingual QE models are effective even with a limited number of training instances. This suggests that we can train a single multilingual QE model on as many languages as possible and apply it to other language pairs. These findings can be beneficial to perform QE in low-resource languages for which the training data is scarce and when maintaining several QE models for different language pairs is arduous. Considering the benefits of multilingual models, we have released several multilingual sentence-level and word-level pre-trained models on HuggingFace model hub.

The main limitation of our multilingual evaluation is that all the languages we used throughout the experiments are supported by the pre-trained XLM-R model we used. XLM-R large model only supports 100 languages at the moment, and there is a lot of low resource but common languages such as Chewa, Tajiki, Tigrinya[2] etc. that XLM-R does not support. A question can arise about how the language pairs that XLM-R does not support perform in our multilingual QE environment. However, as far as we know, until very recently, there were no annotated QE datasets either for the languages outside the 100 languages supported by XLM-R. Therefore, it would not be possible to carry out a proper evaluation. Very recently, in WMT 2021, an annotated QE dataset for Pashto-English and Khmer-English was introduced. XLM-R does not support Pashto and Khmer at the moment, and it would be interesting to experiment with them with our multilingual models, which we hope to do in future work.

Pre-trained multilingual transformer models are rapidly increasing in popularity in the NLP community. From them, one notable multilingual transformer model is mT5 (Xue et al., 2021); multilingual text to text transformer model, which considers every task as a sequence to sequence task. It has provided very good results in a variety of multilingual NLP tasks. As future work, we hope to incorporate mT5 in *TransQuest* framework and evaluate it in a multilingual QE

[2]Chewa, Tajiki and Tigrinya are the official languages of Zimbabwe, Tajikistan and Eritrea respectively that are collectively spoken by more than 30 million people in the world

environment.

With this, we conclude Part III of the thesis, using deep learning based STS metrics in translation quality estimation. We showed that the state-of-the-art STS methods we experimented in Part I of the thesis can be employed successfully in QE. Our method outperform current neural QE models such as OpenKiwi and DeepQuest in word-level and sentence-level QE setting a new state-of-the-art. Furthermore, they are simple compared to the complex neural architectures employed in QE in recent years. We believe that the findings of Part III of the thesis would open a new direction in QE.