Chapter 3

Extending TransQuest for word-level QE

Translation quality can be estimated at different levels of granularity: word, sentence and document level [168]. So far in this thesis, we have only explored sentence-level QE [160], 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 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 we mentioned before, sentence-level QE models are difficult to explain as they just outputs a single score for the translation. Users would face the difficulty of interpreting the score and deciding the steps after the QE process. However, in contrast to that word-level QE models provide more fine-grained information about the quality of a translation and as a result, they can

improve the explainability of the whole QE process. Since there is a growing interest in the NLP community for the explainable machine learning, we believe that having explainable QE models would improve the popularity of QE.

With these advantages researches 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 1, most of the open source quality estimation frameworks like OpenKiwi, uses same architectures for both word-level and sentence-level QE. For example the predictor-estimator architecture used in OpenKiwi provides best results for word-level QE too. Therefore word-level QE also suffers from the same limitation that we mentioned in 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.

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 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 retain word-level qualities. This architecture is very simple and effective when compared with the current state-of-the-art word-level QE architectures like predictor-estimator. We believe that a simpler architecture like that would improve the popularity of word-level QE in real-world applications.

As we mentioned in Chapter 1 in Part III of this thesis, 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 quality of the gaps in the target sentence. For example Marmot [181] only supports predicting the quality of the words in target and OpenKiwi [166] only supports predicting the quality of the words in source and target. They completely ignore predicting the quality of the gaps. Despite that, predicting the quality of the gaps in 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 quality of the words in target, predicting quality of the words in source and predicting quality of the gaps in target. We believe an approach that supports every feature in word-level QE would be stronger than existing solutions.

We address four research questions in this chapter:

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

RQ2: Does crosslingual embeddings have an advantage over multilingual embeddings in the word-level QE task too?

RQ3: Can the proposed model 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 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 [181] and OpenKiwi [166].

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

4. The initial findings of this chapter are published in Ranasinghe et al. [182].

5. We integrated the architecture with TransQuest, which already had two sentence-level architectures we described in Chapter 2[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 3.1 discusses the methodology and the experiments done with eight language pairs in word-level QE. Section 3.2 shows the results and perform the evaluation on three evaluation

[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

metrics uses in word-level QE. Section **??** provide further fine-tuning strategies to improve the results. The chapter finishes with conclusions and ideas for future research directions in word-level QE.

3.1 Methodology

The proposed architecture for the word-level QE is very similar to the *MonoTransQuest* architecture in Chapter 2 where the source and the target is processed through a single transformer model. The difference here is since we need quality for each individual word, we do not use a pooling mechanism like we did with *MonoTransQuest*. Instead we keep the state of the individual tokens to get their quality.

As we need to consider the quality of the GAPs too 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 in to a single transformer. A simple linear layer is added on top of word and <GAP> embeddings to predict whether it is "Good" or "Bad" as shown in Figure 3.1. 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 smallest unit possible in QE, we named the proposed word-level architecture as

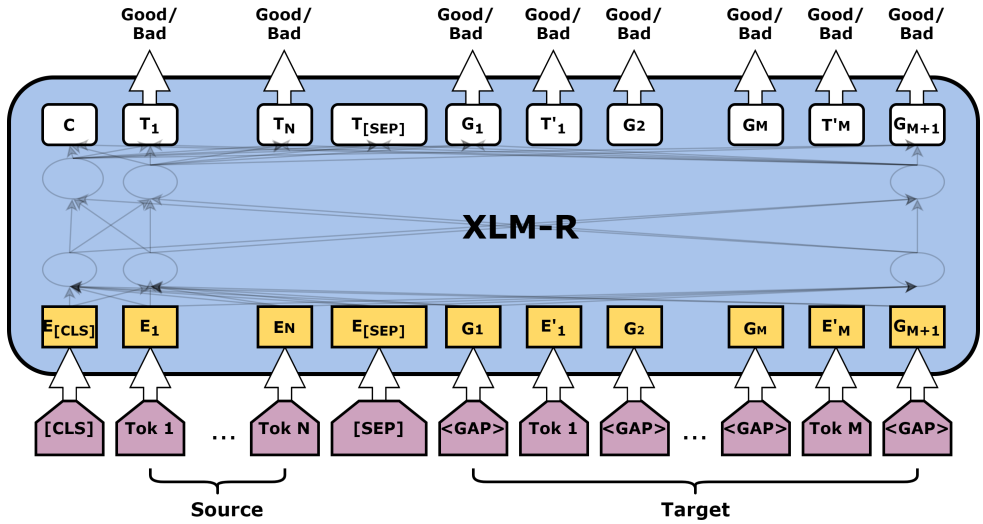

Figure 3.1: MicroTransQuest Architecture

MicroTransQuest.

Similar to the sentence-level QE architectures, we used crosslingual transformer models for this architecture too. As we explained in Chapter 2 XLM-R provides state-of-the-art crosslingual transformer models. There are two pre-trained XLM-R models released by HuggingFace's Transformers library [155]; 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 [176], making it potentially very useful to estimate the word-level translation quality for a large number of language pairs.

3.1.1 Experimental Setup

We evaluated the architecture in word-level quality estimation using the data introduced in Chapter 1. We applied the same set of configurations for all the

[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

language pairs in order 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 learning rate 1e−5, and a linear learning rate warm-up over 10% of the training data. During the training process, the parameters of XLM-R 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 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.

3.2 Results and Evaluation

For 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) [158] as we explained in Chapter 1 in Part III of the thesis. 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

	Method	Mid-resource				High-resource			
		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 3.1: F1-Multi Target between the algorithm predictions and human annotations. Best results for each language by any method are marked 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 3.3. 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.

shared task.

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 3.1, 3.2 and 3.3 respectively. Prior to 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 3.2. For the language pairs in WMT 2019 and 2020 we report a single result for target gaps and words as (F1-Multi Target) in Table 3.1.

Raw I in Tables 3.1, 3.2 and 3.3 shows the results for MicroTransQuest with

	Method	Mid-resource			High-resource	
		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 3.2: F1-Multi GAPS between the algorithm predictions and human annotations. Best results for each language by any method are marked 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 3.3. 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.

XLM-R-large and Raw II in Tables 3.1, 3.2 and 3.3 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 3.1, *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 3.1 also gives the results of the best system submitted for a particular language pair.

	Method	Mid-resource				High-resource			
		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 3.3: F1-Multi Source between the algorithm predictions and human annotations. Best results for each language by any method are marked 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 3.3. 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.

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 GAP evaluation metric in word-level QE, as shown in Table 3.2 *MicroTransQuest* outperforms best systems submitted to the respective shared tasks in all the language pairs we experimented. *MicroTransQuest* achieves ≈ 0.05-0.2 F1-Multi score boost over best systems. Notably *MicroTransQuest* achieves ≈ 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 target sentence. Since *MicroTransQuest*

supports this and produces state-of-the-results in this evaluation metric too, we believe that using *MicroTransQuest* in word-level QE would be beneficial than using OpenKiwi and Marmot.

Finally in F1-Multi Source evaluation metric too, as shown in Table 3.3 *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 source. On the other hand, OpenKiwi supports this, but still *MicroTransQuest* achieves $\approx$ 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 STS architecture be used to predict all features in word-level QE task by just modifying the embeddings and the output layer? We show that state-of-the-art STS method based on crosslingual transformer models can be used in word-level QE too by just changing the output layer and it outperforms current state-of-the-art word-level QE methods like 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 3.1, 3.2 and 3.3 shows the results of multilingual BERT

(mBERT) in MicroTransQuest architecture. 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. As shown in row II in Tables 3.1, 3.2 and 3.3 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 to the results. This observation is similar to what we experienced in Chapter 2 with sentence-level QE experiments.

With this we can answer our **RQ2:** Using crosslingual embeddings with STS architecture proved advantageous rather than using just 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 quality of the gaps in target, we show that XLM-R can be used in tasks similar to detecting missing words too.

3.3 Further Improvements

In this section. we try improve the results of MicroTransQuest through ensemble learning as we did with sentence-level quality estimation. As me mentioned before in Section 3.1 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 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 best results when we used the weights 0.6:0.4. We report the results from this step in row III of Tables 3.1, 3.2 and 3.3 as MicroTransQuest ⊗.

As shown in Tables 3.1, 3.2 and 3.3 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 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.

3.4 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 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 target sentence. The architecture we proposed in this Chapter is very simple when compared with the predictor-estimator architecture in OpenKiwi, yet this architecture produce better results. We further improved the results with ensemble learning showed that *MicroTransQuest* outperforms majority of the best systems submitted for that language in each shared task. We can conclude that state-of-the-art STS architecture can be used in word-level QE too by doing small modifications to the output layers.

The implementation of the architecture, which is based on PyTorch [154] and Hugging Face [155], has been integrated into the TransQuest framework [171] which won the WMT 2020 QE task [160] on sentence-level direct assessment [172][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 word-level and sentence-level qualities separately would be chaotic in some situations. Therefore, in the future we hope to explore multi-task learning for word-level and sentence-level QE [183]. 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 at the same time can be advantageous.

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

As we discussed at multiple points in this thesis, pre-trained models based on transformers are big in size. Therefore, managing a several of them for each language pair would be chaotic in a real-world application for word-level QE too. As a solution for that we explore multilingual word-level QE models in Chapter 4 in Part III of the thesis.