applied sciences

MDPI

Article

Reusing Monolingual Pre-Trained Models by Cross-Connecting Seq2seq Models for Machine Translation

Jiun Oh and Yong-Suk Choi *

Artificial Intelligence Laboratory, Hanyang University, Seoul 04763, Korea; jiunoh@hanyang.ac.kr
* Correspondence: cys@hanyang.ac.kr

Abstract: This work uses sequence-to-sequence (seq2seq) models pre-trained on monolingual corpora for machine translation. We pre-train two seq2seq models with monolingual corpora for the source and target languages, then combine the encoder of the source language model and the decoder of the target language model, i.e., the cross-connection. We add an intermediate layer between the pre-trained encoder and the decoder to help the mapping of each other since the modules are pre-trained completely independently. These monolingual pre-trained models can work as a multilingual pre-trained model because one model can be cross-connected with another model pre-trained on any other language, while their capacity is not affected by the number of languages. We will demonstrate that our method improves the translation performance significantly over the random baseline. Moreover, we will analyze the appropriate choice of the intermediate layer, the importance of each part of a pre-trained model, and the performance change along with the size of the bitext.

Keywords: natural language processing; transfer learning; neural machine translation

Citation: Oh, J.; Choi, Y.-S. Reusing Monolingual Pre-Trained Models by Cross-Connecting Seq2seq Models for Machine Translation. *Appl. Sci.* **2021**, *11*, 8737. https://doi.org/10.3390/app11188737

Academic Editor: Francisco García-Sánchez

Received: 14 August 2021
Accepted: 14 September 2021
Published: 19 September 2021

1. Introduction

Transfer learning with pre-training and fine-tuning has pushed the state-of-the-art results in many natural language processing (NLP) tasks since the famous success of BERT [1]. This method now has become a common practice in the NLP field. The unsupervised pre-training on unlabeled data is particularly beneficial when the labeled training data are limited. The neural machine translation (NMT) task is the case because parallel corpora are expensive to construct and not easily available for many languages, while monolingual corpora are relatively easier to find and large in scale. Therefore, we expect a translation model's performance to be improved by pre-training using rich monolingual data and fine-tuning with parallel data.

However, when using transfer learning for NMT, there are two issues. First, NMT is a cross-lingual task, which involves at least two languages. Recent approach for the cross-lingual tasks is to pre-train a model with multilingual objective [2–4], but there are also several monolingual pre-trained models of various languages, such as [5] for Brazilian [6,7], for Dutch [8], for Romanian [9], for Russian [10–12], for French [13,14], for Italian, and [15] for Portuguese. In multilingual models, each language has only a limited allocation for model capacity, unlike in monolingual models. Furthermore, the monolingual models are relatively easier to create. Therefore, it would be beneficial if the monolingual models can be reused for the machine translation task.

Another issue is that NMT is a sequence generation task. Natural language generation (NLG) tasks, including NMT, usually need both the encoder and the decoder. It can be thought that both the pre-trained encoder and decoder would improve the performance of a language generation model. However, reuse of a whole pre-trained sequence-to-sequence (seq2seq) [16] model is not very common. Existing approaches usually leverage only a part of a seq2seq model, namely an encoder such as BERT and XLM [17], and a decoder such as GPT [18].

Our purpose is to reuse the monolingual seq2seq models pre-trained for monolingual tasks, such as classification or summarization, also for the machine translation task. Yet, a monolingual pre-trained seq2seq model is not fully appropriate for NMT since it is a cross-lingual task. For this, we propose the 'cross-connection', a simple method that use monolingual pre-trained seq2seq models for NMT.

In this paper, we pre-train whole seq2seq models with monolingual corpora, then we construct a translation model by 'cross-connecting' a seq2seq of one language with another seq2seq of a different language, and then we fine-tune the translation model with a parallel corpus. For pre-training, each model receives inputs where some tokens are dropped, and it is trained for each language to predict the dropped tokens. For fine-tuning, a seq2seq model is initialized by the encoder of the source language model and the decoder of the target language model, and it is fine-tuned on parallel data for translation. A remaining problem is that the encoder and the decoder are pre-trained completely separately for different languages, and do not know which language they would be composed with. For mediation between these two modules, we insert an additional 'intermediate layer'. This layer helps mapping of the independently pre-trained encoder and decoder.

Because they are composable, our monolingual pre-trained models can work as a multilingual pre-trained model. A seq2seq model pre-trained on one language can be cross-connected with another model that is pre-trained on any other language. For example, if we have three pre-trained seq2seq models of English, Romanian, and German, we can reuse them to the English-German, English-Romanian, German-Romanian translation models. If we train another seq2seq model on Italian, we can reuse them to the English-Italian, German-Italian, Romanian-Italian translation models. It is possible to create $n(n-1)$ translation pairs with n encoders and decoders. Most existing multilingual pre-trained models such as XLM-R [3] and mBART [2] use a fixed shared vocabulary, which can learn only the languages already in the vocabulary. They are less extensible as both the vocabulary and embedding need to be learned again when adding a new language. Although [19] extended [2] by adding new embedding layers, they also used the fixed shared vocabulary from [3]. In contrast, our method uses separate vocabularies and the models are pre-trained independently. Additionally, according to [3], as the number of languages increases, the per-language capacity decreases. They showed that the size of the model and vocabulary should be larger to improve the performance of a multilingual pre-trained model. However, our method does not need such a massive model size, and the capacity of a pre-trained model is not affected by the number of languages.

We will demonstrate that our cross-connection method considerably improves the performance over the random baseline, and the intermediate layer gives further improvement. Transformer [20] is used as the base seq2seq architecture. For pre-training, the English and German corpora of WMT17 English-German, the Romanian corpus of ParaCrawl English-Romanian, and the Italian corpus of PacraCrawl English-Italian are used. For fine-tuning the IWSLT14 English-German, IWSLT14 English-Romanian, IWSLT14 English-Italian, IWSLT17 Romanian-Italian, IWSLT17 Italian-German, and IWSLT17 German-Romanian corpora are used. The effect of size and configuration of the intermediate layer, the influence of each pre-trained module, and the initialization of cross-attention of the decoder will be analyzed. We will also compare the performance changes along with the different sizes of the fine-tuning datasets.

2. Related Work

2.1. Unsupervised Pre-Training

There are several studies that leverage pre-training and fine-tuning method for NLP tasks in recent years. ELMo [21] proposes pre-trained contextualized representations. GPT [18] uses the language modeling objective and BERT [1], RoBERTa [22], SpanBERT [23] use the MLM objective to pre-train the model on the large monolingual corpora. These models pre-train only the encoder or the decoder of a seq2seq model, which are not suitable architecture for natural language generation. There are other works that propose

pre-training for language generation tasks. XLM [17] pre-trains the Transformer with MLM objective on multiple languages. MASS [24] focuses on joint learning of encoder and decoder for natural language generation. BART [25] applies the denoising autoencoder method to the Transformer and proposes several denoising objectives. UNILM [26] and UniLMv2 [27] proposes the unified modeling that can be applied to both understanding and generation tasks. T5 [28] extensively studies transfer learning in NLP by transforming all tasks to a sequence-to-sequence problem. Ref. [29] explored many unsupervised pre-training objectives and systematically analyzed them in both supervised and unsupervised settings.

2.2. Reusing Pre-Trained Models

Developing a new pre-training objective is important, but it is also necessary to leverage the existing pre-trained models. Our purpose is to reuse the pre-trained monolingual models for machine translation. From this perspective, the work of [30] is the closest to ours that leverages the pre-trained BERT and GPT checkpoints for language generation tasks. However, their translation experiment is done only on the WMT14 En-De pair, and there is no additional layer unlike ours. Furthermore, they use the multilingual BERT to initialize the encoder, but we use monolingual pre-trained encoders. In addition, there are other works such as [31] that fuses BERT into an NMT model by feeding it into both the encoder and decoder, and [32] that proposes a training framework that integrates pre-trained language models to NMT to solve the forgetting problem in resource-rich translation, and [33] that recycles the BERT by replacing the Transformer encoder with BERT, and [34] that studies the various ways to use pre-trained BERT to NMT model and assesses translation quality. On reusing BERT for NMT, Ref. [35] analyzed the difference between the representations that BERT and NMT encoder create and proposed a supervised transformation method. In the field of computer vision, Ref. [36] shares similar concept with ours for the image translation task.

2.3. Multilingual Language Models

Our monolingual pre-trained models can be used as a multilingual pre-trained model by cross-connection. Ref. [37] proposes a multiway, multilingual translation model, which has multiple encoders and decoders with single shared attention. Ref. [38] perform extensive experiments in training massively multilingual NMT models. Ref. [39] adopts an explicit interlingua that converts language-specific embeddings to common representations, which looks similar to ours, yet the role of interlingua is opposite because our intermediate layer does not appear in the pre-training stage. Currently, recent works usually focus on making a single model handle multiple languages. XLM-R [3] is a multilingual masked language model pre-trained in massive scale for cross-lingual understanding. mBART [2], Ref. [19] is a multilingual seq2seq denoising autoencoder for NMT pre-trained with BART objective. MARGE [40] is trained to reconstruct the target text in one language by retrieving related texts in other languages. mT5 [4] inherits T5 and extends it to the multilingual setting. mT6 [41] proposes new cross-lingual pre-training objectives. However, these works use a shared vocabulary which limits the extensibility of multilingual language models and needs a large model size. Ref. [42] demonstrates that the important factor is the effective vocabulary size rather than the subwords sharing. They learn monolingual representation with MLM objective and transfers the knowledge to another language by learning a new embedding matrix while freezing other layers. In line with this, our method excludes the shared vocabulary and trains the models on their respective monolingual data.

3. Method

3.1. Monolingual Unsupervised Pre-Training

The purpose of pre-training is to train the encoder and decoder to be 'fluent' in the corresponding language. We expect the pre-trained model to learn some general knowledge about the language.

In the sequence-to-sequence architecture, the encoder is responsible for natural language understanding (NLU) and the decoder is responsible for natural language generation (NLG). The encoder creates the representation of a given input sequence, and the decoder generates the target sequence conditioned on the encoder representation.

Unlike encoder-only pre-training models such as [1,3], we need the decoder to generate the translated sequence because MT is an NLG task. Therefore, we pre-train the whole sequence-to-sequence models consisting of the encoder and decoder. For the basic bilingual translation system, we train two models for the source and target languages, respectively.

We use the denoising objective for unsupervised learning as in other works. The noise function is simple token dropping which is mentioned in [28]. We choose this method to keep the training simple and focus on the effect of cross-connection itself. 25% of the input tokens are dropped randomly and the model is trained to predict only the dropped tokens by minimizing the cross-entropy. Our implementation is based on the official BERT pre-training code with the whole-word masking algorithm. Please note that any other unsupervised objectives can be used, but we leave this for future work. The architecture of monolingual pre-training is illustrated in Figure 1.

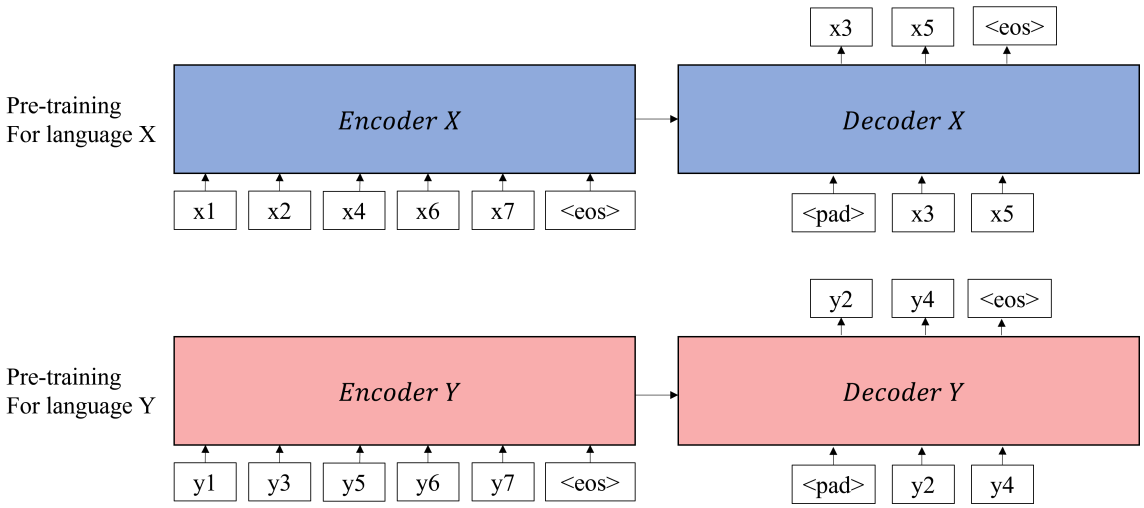


Figure 1. Monolingual pre-training of each model. The model is trained to generate the dropped tokens.

We use the separate disjoint vocabularies for all the languages to encourage the model to reach full capacity for its language. As noted by [42], when using a shared vocabulary for multiple languages, each language cannot occupy the whole vocabulary and has a limited allocation. The vocabulary is inevitably split into portions for all languages, and as the number of the languages grows, each portion becomes smaller. On the other hand, the effect of subwords sharing is not very crucial, because not every subword is effectively shared. To alleviate this problem, the size of the model and vocabulary should be large as shown in [3,42]. However, we do not have these problems because each language model can have its own vocabulary without any need for sharing. Each language occupies the whole vocabulary, and the per language capacity is not affected by the number of languages. The fluency of each pre-trained model is preserved intact.

3.2. Cross-Connection and Fine-Tuning

We compose the pre-trained models for fine-tuning. We reuse the two pre-trained models for the source and target language, respectively. The encoder is initialized by the weights of the encoder from the source language pre-trained model, and the decoder is initialized by the weights of the decoder from the target language pre-trained model. We call this the 'cross-connection'. Then the model is fine-tuned with a parallel corpus for the translation objective. Although the encoder and decoder are not trained for translation, they

can be adapted for translation because the fine-tuning itself achieves implicit alignment as noted in [35].

Each pre-trained model has its expertise in a particular language. Then each language component can be freely combined with any other language component. Once we pre-train multiple seq2seq models and obtain multiple pre-trained encoders and decoders, this enables the fine-tuning for all combinations of the languages among them. We can pick the encoder and decoder for each language and put them together just like blocks. By this, we can exploit the full capacity of the models not losing any ability learned from their pre-training. The architecture of cross-lingual fine-tuning is shown in Figure 2 and the illustration of cross-connection is shown in Figure 3. When translating from language X to language Y, the encoder X and the decoder Y are used. When translating from language Y to language X, the encoder Y and the decoder X are used.

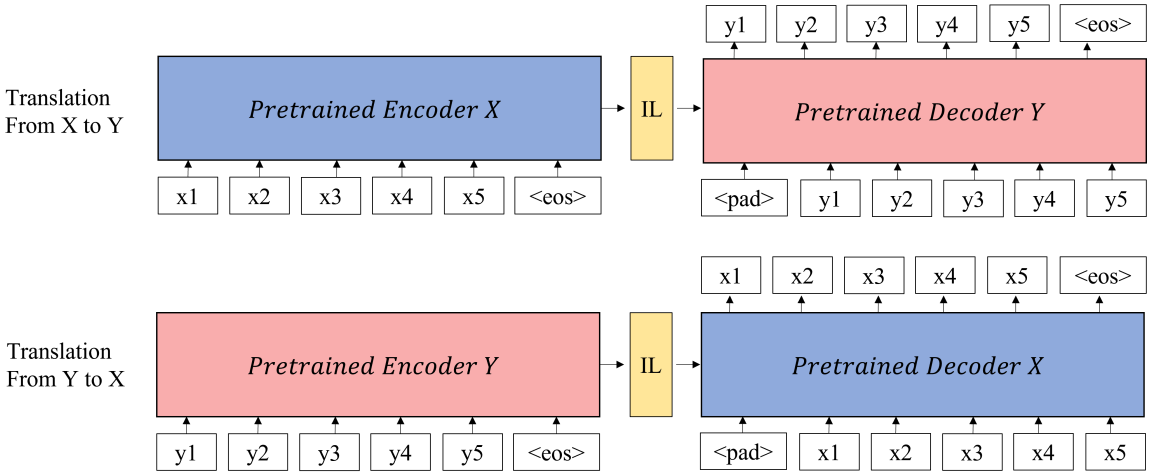


Figure 2. Fine-tuning of the cross-connected model. IL means the additional intermediate layer. We initialize the encoder with the source language pre-trained encoder and the decoder with the target language pre-trained decoder.

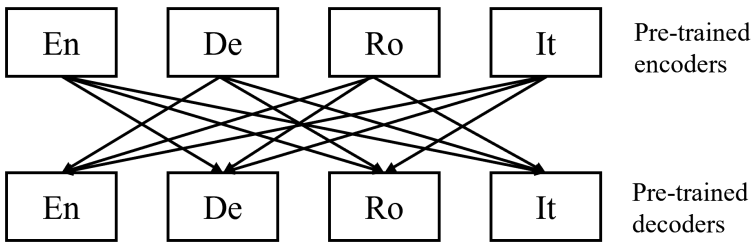


Figure 3. Illustration of cross-connection among multiple pre-trained encoders and decoders. The intermediate layer is omitted for simplification. When creating a translation model, we can pick and compose a proper module for each language among the pre-trained encoders and decoders. Thus, we can use the composition of monolingual models such as a pseudo-multilingual pre-trained model.

3.3. Intermediate Layer

Since the encoder and decoder have been pre-trained independently for different languages, it cannot be guaranteed that the model is suitable for cross-lingual tasks. The encoder knows the source language well, also the decoder knows the target language well, but it is not certain that the combined encoder-decoder model can actually perform the translation well, which is a cross-lingual task.

Therefore, we insert an additional intermediate layer between the encoder and decoder. This layer works as a mediator between the monolingual pre-trained encoder and decoder. The additional parameters help the components to adapt to each other. This is similar to the 'feature mapping layer' from [36] that fills the gap between the representations, which

the encoder is pre-trained to generate and the decoder is pre-trained to reconstruct from. It can be said that this layer maps the separately pre-trained language spaces.

The best configuration of the intermediate layer may vary depending on the size of the model and dataset. We consider the feed-forward network, a self-attention layer, a sublayer of the encoder from the Transformer architecture, and a simple dense layer as the intermediate layer. The feed-forward network consists of two dense layers with ReLU activation, dropout, and layer normalization. The sublayer of the encoder consists of a self-attention layer and a feed-forward network. We will further investigate the effect of configuration change of the intermediate layer in Section 4.5.1.

This layer looks similar to the interlingua of [39], but our approach is different. The purpose of interlingua is to convert language-specific embeddings to language-independent embeddings. It learns all the languages during the training and creates common representation across the languages. However, our intermediate layer does not appear in the pre-training stage but is tailored to each fine-tuning language pair, only helping the mapping of pre-trained encoder and decoder as an intermediator.

4. Experiment

4.1. Architecture and Datasets

We use the standard Transformer model with the Tensorflow official 2.3.0 implementation [43]. We adopt a Transformer-Base architecture with 6-layers of encoder and 6-layers of decoder. The hidden size, number of attention heads, feed-forward filter size is 512, 8, 2048, respectively. The intermediate layers are different for each language pair and the best score is reported among the experiments described in Section 4.5.1 (∼58 M params). We tie the weights of target embeddings and output layer.

For pre-training of English and German, we use the WMT17 [44] En-De. newstest2013 is used as the validation set. For pre-training of Romanian, the Romanian corpus from Paracrawl [45] En-Ro is used. We use newsdev2016 from WMT16 [46] En-Ro as validation set. For pre-training of Italian, we use the Italian corpus from Paracrawl En-It as the training set and the 1% split of the training data as the validation set.

For fine-tuning, we use IWSLT14 [47] En-De, En-Ro, En-It, and IWSLT17 Ro-It, It-De, De-Ro datasets. 7K split of training data are used for the validation of IWSLT14, and dev2010 is used for validation of IWSLT17. For test, we use the concatenation of dev2010, dev2010, tst2010, tst2011, tst2012 for IWSLT14 and tst2010 for IWSLT17. Our preprocessing scripts are adapted from Fairseq [48] translation examples.

We use the Wordpiece algorithm [49] for tokenization and the vocabulary size is 10K per language. The size of the datasets is shown in Table 1. Please note that our pre-training data are relatively small compared to other works due to the limited computational resources.

Table 1. Size/MB of each pre-training and fine-tuning dataset.

Pre-training	WMT17		ParaCrawl			
	En	De	Ro		It	
	639	714	264		1667	
Fine-tuning	IWSLT14			IWSLT17		
	En-De	En-Ro	En-It	It-Ro	It-De	De-Ro
	29.6	30.3	30.5	44.7	45.6	44.3

4.2. Training and Decoding

1 GPU is used for both pre-training and fine-tuning. The batch size is 4096 and the max sequence length is 256. We use Adam optimizer and rsqrt decay scheduling. The hyperparameters are set as $\beta_1 = 0.9$, $\beta_2 = 0.997$, $\epsilon = 1 \times 10^{-9}$, warmup step = 16,000, initial learning rate = 2.0 for pre-training, and $\beta_1 = 0.9$, $\beta_2 = 0.997$, $\epsilon = 1 \times 10^{-9}$, warmup step = 1600, initial learning rate = 0.5 for fine-tuning. We set the dropout rate as 0.1 for

all experiments. We use beam-search with beam size = 4 and $\alpha = 0.6$ for decoding. Our baseline is a randomly initialized Transformer with the same hyperparameters and training schedule of the fine-tuning models. The final models are selected based on the validation loss. The results are reported in BLEU [50] uncased score.

4.3. Models

We compare the fine-tuning models as follows:

- **Random** a randomly initialized baseline without pre-training.
- **Cross-connected** Our cross-connected models that the encoder is pre-trained with source language and the decoder is pre-trained with target language.
- **Cross-connected + IL** Our cross-connected models with the additional intermediate layer.
- **ENC2ENC** It is possible to use the pre-trained encoder weights to initialize the decoder such as BERT2BERT from [30], because the encoder and decoder of the Transformer are implemented in the same architecture except for the cross-attention of the decoder. We compare this scenario with our cross-connected method to see the effect of the pre-trained decoder. We initialize the encoder with the source language encoder weights, and we initialize the *decoder* with the target language *encoder* weights. In this case, the cross-attention of the decoder is initialized randomly.

4.4. Results

As shown in Table 2, for all datasets, our method has significant improvement over the random baseline, especially +4.27 points for De-En and +4.03 points for It-En. We observe that the most important factor is pre-training itself. The simple cross-connection obtains a large gain of BLEU, which demonstrates that reusing the monolingual pre-trained models by cross-connection can considerably improve the translation performance. When compare ENC2ENC and Cross-connect, ENC2ENC improves over Random but the cross-connected models report higher scores overall. This shows that initializing the decoder with pre-trained encoder weights is useful than random but not optimal. Furthermore, the scores of cross-connection models further improve with the additional intermediate layer, which means that the intermediate layer helps to combine the independently trained encoder and decoder to some extent, although not as crucial as our initial assumption.

Table 2. BLEU scores for the IWSLT14 En-De, IWSLT14 En-Ro, IWSLT14 En-It, and IWSLT17 It-Ro datasets. IL is the intermediate layer and its configuration is different for each model. We indicate the best scores with **bold** font.

Model	**En-De**	**De-En**	**En-Ro**	**Ro-En**	**En-It**	**It-En**
Random (Baseline)	26.11	33.83	27.49	37.47	29.10	31.44
ENC2ENC	28.36	37.16	29.31	39.06	31.18	34.22
Cross-connected	29.06	37.75	30.04	39.73	32.17	35.05
Cross-connected + IL	**29.63**	**38.10**	**30.64**	**40.39**	**32.29**	**35.47**
Model	**It-Ro**	**Ro-It**	**It-De**	**De-It**	**De-Ro**	**Ro-De**
Random	21.73	22.27	18.91	19.70	18.37	19.50
ENC2ENC	22.58	23.64	20.74	22.17	20.94	20.53
Cross-connected	22.36	23.91	21.53	22.65	20.98	21.20
Cross-connected + IL	**23.03**	**24.33**	**22.04**	**23.32**	**21.26**	**21.71**

4.5. Ablation Study

4.5.1. Intermediate Layer

We conduct several experiments to see the effects of different architecture and size of the intermediate layer. We consider the feed-forward network with various filter sizes, a self-attention layer, a single sublayer of the encoder that consists of a self-attention layer

and a feed-forward network with filter size 2048, and a simple dense layer of size 512 with layer normalization and dropout.

The results are shown in Table 3. Overall a feed-forward network reports the best score though the size varies. The single dense layer consistently results in low scores as expected, except for Ro-It. However, more numbers and bigger size of the intermediate layer do not necessarily lead to higher BLEU scores, which indicates that a larger number of parameters does not exactly mean better performance. Further, we assumed that the self-attention mechanism would benefit the mapping of the encoder and decoder, but it does not result in the best score except for the En-Ro experiment which uses a sublayer of the encoder as the intermediate layer. Overall, We recommend a feed-forward network with filter size 512 or 768 because this network usually performs well with relatively fewer parameters than the encoder sublayer or the self-attention layer.

Table 3. Comparisons of the fine-tuning models with various intermediate layers for each translation pair. FFN is the feed-forward network with ReLU activation. Dense is a fully connected layer with layer normalization and dropout. Results below 'No intermediate layer' are sorted by the number of parameters in decreasing order. We indicate the best scores with **bold** font.

Model	En-De	De-En	En-Ro	Ro-En	En-It	It-En
No intermediate layer	29.06	37.75	30.04	39.73	32.17	35.05
2FFN (filter size = 2048)	29.43	38.08	30.31	40.01	31.80	35.14
Encoder Sublayer (filter size = 2048)	29.55	38.00	**30.64**	40.02	32.06	35.20
2FFN (filter size = 1024)	29.20	38.01	30.29	**40.39**	32.09	35.22
1FFN (filter size = 2048)	29.37	**38.10**	30.23	40.21	**32.29**	35.45
1FFN (filter size = 1024)	29.57	37.91	30.37	40.20	32.08	35.23
Self-Attention	29.33	38.03	30.45	40.14	32.01	35.17
1FFN (filter size = 768)	**29.63**	38.01	30.27	40.20	32.02	35.44
1FFN (filter size = 512)	29.52	37.94	30.28	40.27	32.09	**35.47**
Dense (size = 512)	29.18	37.77	30.01	40.20	32.08	35.06
Model	**It-Ro**	**Ro-It**	**It-De**	**De-It**	**De-Ro**	**Ro-De**
No intermediate layer	22.36	23.91	21.53	22.65	20.98	21.20
2FFN (filter size = 2048)	22.61	23.94	21.62	23.19	21.18	**21.71**
Encoder Sublayer (filter size = 2048)	22.72	23.92	21.46	**23.32**	20.69	21.44
2FFN (filter size = 1024)	22.95	24.26	21.29	23.05	21.03	21.32
1FFN (filter size = 2048)	22.96	23.50	21.41	23.17	20.58	21.19
1FFN (filter size = 1024)	22.42	24.16	21.69	23.16	21.05	20.93
Self-Attention	22.39	24.07	21.64	22.96	20.68	21.03
1FFN (filter size = 768)	**23.03**	23.74	21.90	23.01	21.00	21.03
1FFN (filter size = 512)	22.08	24.11	**22.04**	23.28	**21.26**	21.14
Dense (size = 512)	22.24	**24.33**	21.80	22.89	21.02	21.42

4.5.2. Initialization of Decoder Cross-Attention

As we mentioned earlier, we can use the pre-trained encoder weights to initialize the decoder. Furthermore, since our method pre-trains each model with only monolingual data, the decoder cross-attention could not be suitable for the cross-lingual task. Therefore, we conduct two additional experiments to investigate the initialization of the decoder. First, we use the pre-trained decoder weights but initialize the cross-attention randomly (Random cross-attention). Second, we use the pre-trained encoder weights to initialize both the encoder and decoder as in Section 4.4 and add the intermediate layer (ENC2ENC). We compare these settings with our cross-connected models (ENC2DEC) and random baselines (RND2RND).

As shown in Table 4, the cross-connection models result in the best scores. The scores decrease when we initialize the cross-attention randomly, and the ENC2ENC with IL models report similar or lower scores than the random cross-attention models. Therefore, it can be said that the monolingual pre-trained decoder can be more effectively reused than the pre-trained encoder for machine translation. Although its cross-attention never

learned cross-lingual tasks, it is better than the simple random initialization. Furthermore, the score gap between the ENC2ENC+IL and ENC2DEC+IL is bigger than the score gap of ENC2ENC and ENC2DEC without IL in Section 4.4. This indicates that the intermediate layer is more effective for the cross-connected model than for ENC2ENC.

Table 4. Fine-tuning models with different decoder initialization. The Baseline is a randomly initialized Transformer and IL is the intermediate layer. We indicate the best scores with **bold** font.

Model	En-De	De-En	En-Ro	Ro-En	En-It	It-En
RND2RND (baseline)	26.11	33.83	27.49	37.47	29.10	31.44
ENC2DEC+IL	**29.63**	**38.10**	**30.64**	**40.39**	**32.29**	**35.47**
ENC2DEC+IL (Random Cross-attention)	29.24	37.90	30.14	39.72	31.81	35.03
ENC2ENC+IL	28.68	37.19	29.73	38.98	31.34	34.22
Model	**It-Ro**	**Ro-It**	**It-De**	**De-It**	**De-Ro**	**Ro-De**
RND2RND (baseline)	21.73	22.27	18.91	19.70	18.37	19.50
ENC2DEC+IL	**23.03**	**24.33**	**22.04**	**23.32**	**21.26**	**21.71**
ENC2DEC+IL (Random Cross-attention)	22.65	23.84	21.32	22.71	20.91	21.14
ENC2ENC+IL	22.71	23.43	20.33	22.01	20.80	20.76

4.5.3. Importance of Each Module

We perform some analysis to show the importance of each module of the pre-trained model in fine-tuning. We compare four models: a random baseline (RND2RND), a model with the pre-trained encoder and random decoder (ENC2RND), a model with random encoder and pre-trained decoder (RND2DEC), and our cross-connected model (ENC2DEC). We do not add the intermediate layer to the ENC2RND or the RND2DEC model because there is no need for mediation, and we neither use it to the cross-connected ENC2DEC model for fair comparison.

The results are reported in Table 5. As expected, when reusing a part of a pre-trained model, the encoder is much more important than the decoder. Both the pre-trained encoder and pre-trained decoder improves the model performance, but the ENC2RND model obtains a larger gain. This seems natural because the decoder depends on the source information from the encoder side, and the performance of a sequence-to-sequence model is heavily affected by the sentence representation that the encoder generates [51]. This result suggests that if only a part of a pre-trained model can be reused, it is more efficient to reuse the encoder part than the decoder part. However, the scores of the ENC2DEC models are the highest except for It-Ro, which indicates that cross-connection is more effective than reusing only the encoder.

Table 5. BLEU scores of the fine-tuning models with partial module reusing. We indicate the best scores with **bold** font.

Model	En-De	De-En	En-Ro	Ro-En	En-It	It-En
RND2RND (baseline)	26.11	33.83	27.49	37.47	29.10	31.44
ENC2RND	28.35	36.63	29.63	39.22	30.66	33.80
RND2DEC	27.13	34.88	28.01	37.86	29.81	32.10
ENC2DEC	**29.06**	**37.75**	**30.04**	**39.73**	**32.17**	**35.05**
Model	**It-Ro**	**Ro-It**	**It-De**	**De-It**	**De-Ro**	**Ro-De**
RND2RND (baseline)	21.73	22.27	18.91	19.70	18.37	19.50
ENC2RND	**22.69**	23.14	20.64	22.58	20.43	20.85
RND2DEC	20.59	22.27	19.40	19.96	18.94	20.24
ENC2DEC	22.36	**23.91**	**21.53**	**22.65**	**20.98**	**21.20**

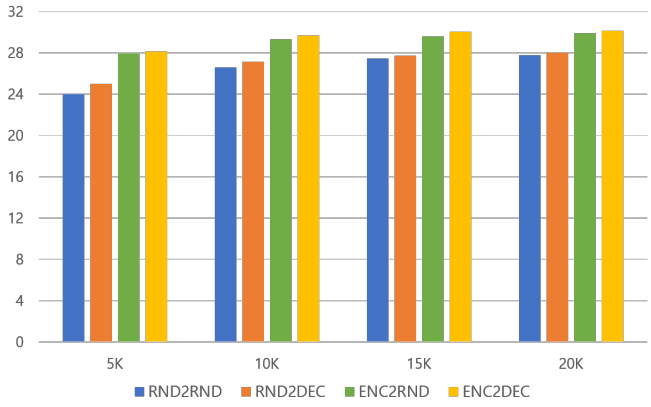
4.5.4. Low-Resource and Mid-Resource

The IWSLT14 and IWSLT17 datasets are low-resource datasets. We perform additional experiments to see if the cross-connection method is also effective for a larger dataset. We use WMT16 En-Ro and IWSLT14 En-Ro to exclude the linguistic characteristics and only compare the size of the fine-tuning datasets. For fine-tuning on the WMT16 En-Ro dataset, we use the same pre-trained weights and different training schedule: initial learning rate = 2.0 and warmup steps = 16,000.

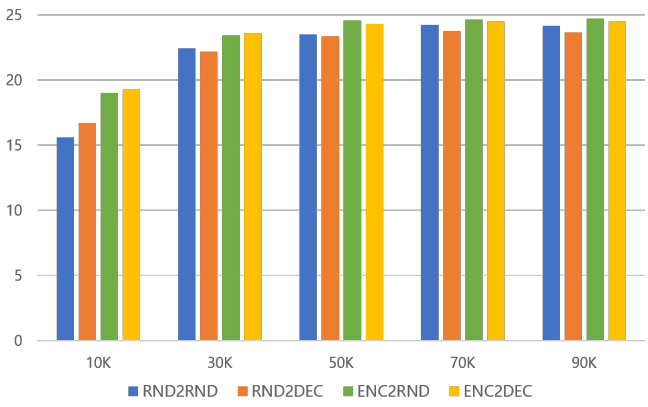
The results are reported in Table 6. We observe that the improvement gap becomes smaller in WMT16 En-Ro as shown in the top section. We conjecture that this result is mainly because of the relatively small amount of our pre-training. However, it is also consistent with the results of other works such as [2,28] that the effect of pre-training decreases or is lost as the size of the parallel data grows. In the middle section, WMT16 ENC2RND obtains the best score and RND2DEC reports a lower score than the random baseline, contrary to IWSLT14. To investigate this further, we illustrate the test scores along with the training steps in Figure 4. WMT16 RND2DEC is outperformed by the RND2RND baseline after training 30K steps. We assume that the effect of cross-connection in a richer dataset is limited by this forgetting of decoder pre-training. It seems to be the reason the ENC2RND obtains a better score than ENC2DEC in WMT16 unlike in IWSLT14.

Table 6. BLEU scores of the IWSLT14 En-Ro dataset and WMT16 En-Ro dataset with partial pre-trained module reusing. We indicate the best scores with **bold** font.

Models	IWSLT14	WMT16
RND2RND (baseline)	27.49	23.89
ENC2DEC	**30.04**	24.51
ENC2RND	29.63	**24.77**
RND2DEC	28.01	23.73
RND2ENC	27.41	24.03
ENC2ENC	29.31	24.68



(**a**) IWSLT14



(**b**) WMT16

Figure 4. BLEU scores of the IWSLT14 En-Ro dataset and WMT16 En-Ro dataset along with the training steps.

This is in line with the results of [30] where their multilingual BERT and English GPT model (BERT2GPT) could not reach the highest score in De-En translation. We surmise that their pre-trained GPT forgot its pre-training because WMT14 En-De is a high-resource dataset. Furthermore, their RND2GPT model gets a score even below the random baseline just like our RND2DEC, and BERT2BERT is similar to the BERT2RND and RND2BERT is better than RND2GPT. These are in agreement with our results reported in the bottom section of Table 6. In WMT16 En-Ro, ENC2ENC obtains the second-best score and RND2ENC is better than RND2DEC, whereas RND2DEC is better than RND2ENC

in IWSLT14 En-Ro. When the amount of bitext grows, the performance of the pre-trained decoder is soon outperformed by the random baseline even though it was higher at first. Therefore, we conclude that cross-connection of pre-trained encoder and decoder is effective for low-resource language translation, and when the resource is rich enough, it is rather effective to use the pre-trained encoders to initialize both the encoder and decoder.

5. Conclusions

In this work, we proposed a novel method that efficiently reuses the pre-trained monolingual seq2seq models for machine translation. Our method pre-trains several seq2seq models with monolingual corpora independently, cross-connects them with an additional intermediate layer, and then fine-tunes the translation model on the parallel corpus. Because we can create $n(n-1)$ translation pairs with n encoders and decoders, the monolingual pre-trained models can work as a multilingual pre-trained model. Their capacity is not affected by the number of languages because each model is pre-trained separately. We showed that the translation performance is improved significantly by our cross-connection method. We investigated the size and architecture of the intermediate layer and we found that the bigger parameters of the intermediate layer do not necessarily improve the performance. Furthermore, we showed that random initialization of decoder cross-attention is not optimal. When initializing the decoder, using pre-trained encoder weights is possible but using pre-trained decoder weights is better. The pre-trained decoder contributes to the performance improvement, although its effect is not as crucial as the encoder because the decoder depends on the representations that the encoder generates. Further, we found that the pre-trained decoder loses its effect when the fine-tuning data are plenty, and the cross-connection method is suitable for the low-resource machine translation. Our limitation is that the size of the model and the number of languages are limited due to the computational resources. For future work, we will extend our study to other language pairs and increase the number of languages.

Author Contributions: Conceptualization and methodology, J.O. and Y.-S.C.; investigation, J.O.; data curation and conceiving experiments, J.O.; performing experiments and designing results, J.O.; writing—original draft preparation, J.O.; writing—review and editing, J.O. and Y.-S.C.; supervision, Y.-S.C.; funding acquisition, Y.-S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No.2018R1A5A7059549), National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No.2020R1A2C1014037), and the Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2020-0-01373, Artificial Intelligence Graduate School Program (Hanyang University)).

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: [https://www.statmt.org/, accessed on 14 August 2021, https://sites.google.com/site/iwsltevaluation2014/data-provided, accessed on 14 August 2021, https://sites.google.com/site/iwsltevaluation2017, accessed on 14 August 2021].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
2. Liu, Y.; Gu, J.; Goyal, N.; Li, X.; Edunov, S.; Ghazvininejad, M.; Lewis, M.; Zettlemoyer, L. Multilingual denoising pre-training for neural machine translation. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 726–742. [CrossRef]
3. Conneau, A.; Khandelwal, K.; Goyal, N.; Chaudhary, V.; Wenzek, G.; Guzmán, F.; Grave, E.; Ott, M.; Zettlemoyer, L.; Stoyanov, V. Unsupervised cross-lingual representation learning at scale. *arXiv* **2019**, arXiv:1911.02116.
4. Xue, L.; Constant, N.; Roberts, A.; Kale, M.; Al-Rfou, R.; Siddhant, A.; Barua, A.; Raffel, C. mT5: A massively multilingual pre-trained text-to-text transformer. *arXiv* **2020**, arXiv:2010.11934.

5. Carmo, D.; Piau, M.; Campiotti, I.; Nogueira, R.; Lotufo, R. PTT5: Pretraining and validating the T5 model on Brazilian Portuguese data. *arXiv* **2020**, arXiv:2008.09144.
6. de Vries, W.; van Cranenburgh, A.; Bisazza, A.; Caselli, T.; van Noord, G.; Nissim, M. Bertje: A dutch bert model. *arXiv* **2019**, arXiv:1912.09582.
7. Delobelle, P.; Winters, T.; Berendt, B. Robbert: A dutch roberta-based language model. *arXiv* **2020**, arXiv:2001.06286.
8. Dumitrescu, S.D.; Avram, A.M.; Pyysalo, S. The birth of Romanian BERT. *arXiv* **2020**, arXiv:2009.08712.
9. Kuratov, Y.; Arkhipov, M. Adaptation of deep bidirectional multilingual transformers for russian language. *arXiv* **2019**, arXiv:1905.07213.
10. Martin, L.; Muller, B.; Suárez, P.J.O.; Dupont, Y.; Romary, L.; de La Clergerie, É.V.; Seddah, D.; Sagot, B. Camembert: A tasty french language model. *arXiv* **2019**, arXiv:1911.03894.
11. Le, H.; Vial, L.; Frej, J.; Segonne, V.; Coavoux, M.; Lecouteux, B.; Allauzen, A.; Crabbé, B.; Besacier, L.; Schwab, D. Flaubert: Unsupervised language model pre-training for french. *arXiv* **2019**, arXiv:1912.05372.
12. Louis, A. BelGPT-2: A GPT-2 Model Pre-Trained on French Corpora. 2020. Available online: https://github.com/antoiloui/belgpt2 (accessed on 14 August 2021).
13. Polignano, M.; Basile, P.; De Gemmis, M.; Semeraro, G.; Basile, V. Alberto: Italian BERT language understanding model for NLP challenging tasks based on tweets. In Proceedings of the 6th Italian Conference on Computational Linguistics, CLiC-it 2019, Bari, Italy, 13–15 November 2019; Volume 2481, pp. 1–6.
14. De Mattei, L.; Cafagna, M.; Dell'Orletta, F.; Nissim, M.; Guerini, M. Geppetto carves italian into a language model. *arXiv* **2020**, arXiv:2004.14253.
15. Souza, F.; Nogueira, R.; Lotufo, R. Portuguese named entity recognition using BERT-CRF. *arXiv* **2019**, arXiv:1909.10649.
16. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.
17. Lample, G.; Conneau, A. Cross-lingual language model pretraining. *arXiv* **2019**, arXiv:1901.07291.
18. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving Language Understanding by Generative Pre-Training; 2018. Available online: https://www.cs.ubc.ca/~amuham01/LING530/papers/radford2018improving.pdf (accessed on 14 August 2021).
19. Tang, Y.; Tran, C.; Li, X.; Chen, P.J.; Goyal, N.; Chaudhary, V.; Gu, J.; Fan, A. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv* **2020**, arXiv:2008.00401.
20. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. *arXiv* **2017**, arXiv:1706.03762.
21. Peters, M.E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep contextualized word representations. *arXiv* **2018**, arXiv:1802.05365.
22. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
23. Joshi, M.; Chen, D.; Liu, Y.; Weld, D.S.; Zettlemoyer, L.; Levy, O. Spanbert: Improving pre-training by representing and predicting spans. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 64–77. [CrossRef]
24. Song, K.; Tan, X.; Qin, T.; Lu, J.; Liu, T.Y. Mass: Masked sequence to sequence pre-training for language generation. *arXiv* **2019**, arXiv:1905.02450.
25. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv* **2019**, arXiv:1910.13461.
26. Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; Hon, H.W. Unified language model pre-training for natural language understanding and generation. *arXiv* **2019**, arXiv:1905.03197.
27. Bao, H.; Dong, L.; Wei, F.; Wang, W.; Yang, N.; Liu, X.; Wang, Y.; Gao, J.; Piao, S.; Zhou, M.; others. Unilmv2: Pseudo-masked language models for unified language model pre-training. In Proceedings of the International Conference on Machine Learning, PMLR 2020, Virtual, 13–18 July 2020; pp. 642–652.
28. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv* **2019**, arXiv:1910.10683.
29. Baziotis, C.; Titov, I.; Birch, A.; Haddow, B. Exploring Unsupervised Pretraining Objectives for Machine Translation. In *Findings of the Association for Computational Linguistics, Proceedings of the ACL-IJCNLP 2021, Online, 1–6 August 2021*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 2956–2971. [CrossRef]
30. Rothe, S.; Narayan, S.; Severyn, A. Leveraging pre-trained checkpoints for sequence generation tasks. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 264–280. [CrossRef]
31. Zhu, J.; Xia, Y.; Wu, L.; He, D.; Qin, T.; Zhou, W.; Li, H.; Liu, T.Y. Incorporating bert into neural machine translation. *arXiv* **2020**, arXiv:2002.06823.
32. Yang, J.; Wang, M.; Zhou, H.; Zhao, C.; Zhang, W.; Yu, Y.; Li, L. Towards making the most of bert in neural machine translation. In Proceedings of the AAAI Conference on Artificial Intelligence, 2020, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 9378–9385.
33. Imamura, K.; Sumita, E. Recycling a pre-trained BERT encoder for neural machine translation. In Proceedings of the 3rd Workshop on Neural Generation and Translation, Hong Kong, China, 4 November 2019; pp. 23–31.
34. Clinchant, S.; Jung, K.W.; Nikoulina, V. On the use of bert for neural machine translation. *arXiv* **2019**, arXiv:1909.12744.

35. Vázquez, R.; Celikkanat, H.; Creutz, M.; Tiedemann, J. On the differences between BERT and MT encoder spaces and how to address them in translation tasks. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop, Online, 5–6 August 2021; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 337–347. [CrossRef]
36. Yoo, J.; Eom, H.; Choi, Y.S. Image-To-Image Translation Using a Cross-Domain Auto-Encoder and Decoder. *Appl. Sci.* **2019**, *9*, 4780. [CrossRef]
37. Firat, O.; Cho, K.; Bengio, Y. Multi-way, multilingual neural machine translation with a shared attention mechanism. *arXiv* **2016**, arXiv:1601.01073.
38. Aharoni, R.; Johnson, M.; Firat, O. Massively multilingual neural machine translation. *arXiv* **2019**, arXiv:1903.00089.
39. Lu, Y.; Keung, P.; Ladhak, F.; Bhardwaj, V.; Zhang, S.; Sun, J. A neural interlingua for multilingual machine translation. *arXiv* **2018**, arXiv:1804.08198.
40. Lewis, M.; Ghazvininejad, M.; Ghosh, G.; Aghajanyan, A.; Wang, S.; Zettlemoyer, L. Pre-training via paraphrasing. *arXiv* **2020**, arXiv:2006.15020.
41. Chi, Z.; Dong, L.; Ma, S.; Mao, S.H.X.L.; Huang, H.; Wei, F. mT6: Multilingual Pretrained Text-to-Text Transformer with Translation Pairs. *arXiv* **2021**, arXiv:2104.08692.
42. Artetxe, M.; Ruder, S.; Yogatama, D. On the cross-lingual transferability of monolingual representations. *arXiv* **2019**, arXiv:1910.11856.
43. Yu, H.; Chen, C.; Du, X.; Li, Y.; Rashwan, A.; Hou, L.; Jin, P.; Yang, F.; Liu, F.; Kim, J.; et al. TensorFlow Model Garden. 2020. Available online: https://github.com/tensorflow/models (accessed on 14 August 2021).
44. Bojar, O.r.; Chatterjee, R.; Federmann, C.; Graham, Y.; Haddow, B.; Huang, S.; Huck, M.; Koehn, P.; Liu, Q.; Logacheva, V.; Monz, C.; Negri, M.; Post, M.; Rubino, R.; Specia, L.; Turchi, M. Findings of the 2017 Conference on Machine Translation (WMT17). In Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers, Copenhagen, Denmark, 7–11 September 2017; Association for Computational Linguistics: Copenhagen, Denmark, 2017; pp. 169–214.
45. ParaCrawl. 2018. Available online: https://paracrawl.eu/ (accessed on 14 August 2021).
46. Bojar, O.r.; Chatterjee, R.; Federmann, C.; Graham, Y.; Haddow, B.; Huck, M.; Jimeno Yepes, A.; Koehn, P.; Logacheva, V.; Monz, C.; et al. Findings of the 2016 Conference on Machine Translation. In Proceedings of the First Conference on Machine Translation, Berlin, Germany, 11–12 August 2016; Association for Computational Linguistics: Berlin, Germany, 2016; pp. 131–198.
47. Cettolo, M.; Girardi, C.; Federico, M. Wit3: Web inventory of transcribed and translated talks. In Proceedings of the 16th Conference of European Association for Machine Translation, Trento, Italy, 28–30 May 2012; pp. 261–268.
48. Ott, M.; Edunov, S.; Baevski, A.; Fan, A.; Gross, S.; Ng, N.; Grangier, D.; Auli, M. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In Proceedings of NAACL-HLT 2019: Demonstrations, Minneapolis, MN, USA, 2–7 June 2019.
49. Wu, Y.; Schuster, M.; Chen, Z.; Le, Q.V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv* **2016**, arXiv:1609.08144.
50. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. Bleu: A method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadephia, PA, USA, 7–12 July 2002; pp. 311–318.
51. Dalvi, F.; Durrani, N.; Sajjad, H.; Belinkov, Y.; Vogel, S. Understanding and Improving Morphological Learning in the Neural Machine Translation Decoder. In Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Taipei, Taiwan, 27 November–1 December 2017; Asian Federation of Natural Language Processing: Taipei, Taiwan, 2017; pp. 142–151.