applied sciences

MDPI

Article

Impact of Sentence Representation Matching in Neural Machine Translation

Heeseung Jung [1], Kangil Kim [1,*], Jong-Hun Shin [2], Seung-Hoon Na [3,*], Sangkeun Jung [4] and Sangmin Woo [5]

[1] Gwangju Institute of Science and Technology (GIST), Gwangju 61005, Korea; heesng.jung@gmail.com
[2] Electronics and Telecommunications Research Institute (ETRI), Gwangju 61012, Korea; jhshin82@etri.re.kr
[3] Department of Computer Science, Jeonbuk National University, Jeonju-si 54896, Korea
[4] Computer Science and Engineering, Chungnam National University, Daejeon 34134, Korea; hugman@cnu.ac.kr
[5] Korea Advanced Institute of Science and Technology, Daejeon 34141, Korea; shmwoo9395@gmail.com
* Correspondence: kangil.kim.01@gmail.com (K.K.); nash@jbnu.ac.kr (S.-H.N.)

Abstract: Most neural machine translation models are implemented as a conditional language model framework composed of encoder and decoder models. This framework learns complex and long-distant dependencies, but its deep structure causes inefficiency in training. Matching vector representations of source and target sentences improves the inefficiency by shortening the depth from parameters to costs and generalizes NMTs with a different perspective to cross-entropy loss. In this paper, we propose matching methods to derive the cost based on constant word-embedding vectors of source and target sentences. To find the best method, we analyze the impact of the methods with varying structures, distance metrics, and model capacity in a French to English translation task. An optimally configured method is applied to English translation tasks from and to French, Spanish, and German. In the tasks, the method showed performance improvement by 3.23 BLEU at maximum, with an improvement of 0.71 on average. We evaluated the robustness of this method to various embedding distributions and models, such as conventional gated structures and transformer networks, and empirical results showed that it has a higher chance to improve performance in those models.

Keywords: recurrent neural network; machine translation; similarity; sentence representation; guiding pressure

Citation: Jung, H.; Kim, K.; Shin, J.-H.; Na, S.-H.; Jung, S.; Woo, S. Impact of Sentence Representation Matching in Neural Machine Translation. *Appl. Sci.* **2022**, *12*, 1313. https://doi.org/10.3390/app12031313

Academic Editor: Takayoshi Kobayashi

Received: 6 December 2021
Accepted: 19 January 2022
Published: 26 January 2022

1. Introduction

Most decoders of neural machine translation (NMT) are conditional language models, which sequentially generate target words in the condition of a given source sentence. This approach is a greedy algorithm, so the dependency between sequentially selected target words may restrict the selection of the best target word composition. Beam search is a promising method to approximate the correct compositions. However, inversely, the promising results imply that NMTs are still weak regarding learning the dependency between output words in the model. This limitation in training is a fundamental barrier in learning in an end-to-end NMT model while not relying on an additional model or algorithm in inference, as proposed in various approaches [1,2]. An effective method to relax the limit is to penalize the cross-entropy by adding a sentence-level score [2–4] using various information generated from a decoder. Beyond the improvement of prediction quality in the works, mapping between two sentence representations has more fundamental meaning because it can provide useful information about the unseen difficulty of training in the typical NMT.

In this paper, we propose a *sentence representation matching* method to apply the direct mapping of sentence-level semantics to existing NMT frameworks. This method is

designed for guiding training of a model with constant word vectors. To obtain a more optimized structure and deeper understanding about the guiding pressure generated by the method, we analyze its impact in the framework with varying structures, distance metrics, and model capacity determined by layer dimensions. The best model derived from the analysis is applied to practical translation tasks of English from and to the French, Spanish, German languages. Then, we analyze its robustness to various embedding distributions and model structures as long-short term memory in recurrent neural networks and transformer networks.

The contributions of this paper are that it:

- proposes a sentence representation matching method and loss functions for training;
- provides the impact analysis of structural factors to control guiding pressure;
- provides a good-performing setting in many models, embeddings and translation tasks.

Section 2 shows related and background works first, then Section 3 explains motivations and details of the proposed method and proposes the factors that affect the guiding pressure to test variants of the method. Section 4 shows the experimental setting on data and model for analysis. Section 5 describes their results and provides a discussion. Sections 6 and 7 present the conclusion and discuss future work.

2. Background

Since 2014, there has been a surge of applying recurrent neural networks with long short term memory (LSTM) [5] to machine translation after the possibility of a learning end-to-end translation model was reported [6]. After intensive development across more than two years, this approach became the state-of-the-art of machine translation and was called NMT. The remarkable research which improved the performance of NMT included the bidirectional LSTM using both forward and backward sequences [6–8], attention model to learn explicit alignment models [9,10], rare word modeling to estimate unknown words by an explicit model and alignment model [11], and argumentation methods to overcome a lack of data [12,13].

Those works have been made more rigorous by adopting many advanced methods such as batch normalization [14], ensemble, beam search, input feature specialization, and input feeding. Those techniques are aggregated in Google's NMT report [15].

In 2017, transformer, a feed-forward model with a self-attention mechanism, was suggested and had a remarkable performance which was better than LSTM [16]. More recently, the very deep transformer model demonstrated higher performance than the vanilla transformer [17], and the pre-trained model also demonstrated remarkable performance [18–20].

Direct mapping of semantic vectors between two languages has been studied in various research directions, but the vector distribution is complex in translation tasks, so simple and successful direct mapping models have not been proposed so far. Using recently used complex NMT frameworks, the model-based approach was able to manage most of the complex relations between words. However, the direct mapping approach is still useful to understand the macroscopic similarity between two concepts. In word-level mapping models [21–23], the similarity between semantic vectors has been used as a dictionary to determine the most semantically related words, even though the mapping is still ambiguous for words used for various purposes. The direct mapping models can be understood as an extremely regularized model for translation, compared to current NMT frameworks, which are particularly difficult to regularize because of the high sensitivity of the parameters in the recurrent networks [24]. In this paper, we propose a safe method to use the macroscopic information to guide current NMT frameworks.

A pointer generator network is a model which uses sentence-level semantic matching [25]; it refines the probability vector to select a class from the information of the distributed representation. This network has shown promising improvement, but it is designed only for enhancing performance rather than guiding the internal hypotheses

of NMT to a more natural translation model. In this paper, we focus on injecting the information of the direct mapping model involving knowledge of regularizing and also guiding the NMT.

3. Sentence Representation Matching

3.1. Motivation

Beam search is a promising method for NMTs by overcoming the problem of greedy searches in sequential target word generation. On the other hand, the impact of beam search inversely implies that the sequential decisions by the model are likely to be incorrect to select the best sentences in many cases. There are many possible causes for the inaccurate prediction of the composition of target words, such as inaccurate model representation, complex parameter landscapes, and noisy data.

A possible cause is the simple representation of the correctness of target words. In current NMTs, cross-entropy is the most popular cost function and is composed of probabilities of selecting each correct word of a target sequence. Therefore, only one variable is responsible for representing whether the selected target word is correct. Using only one variable may be risky because the second probable word and its highly probable following sequences may give higher cross-entropy than any sequences derived from the correct word selection. This case is a deceptive example of restricting accurate word composition in decoders.

Another cause is the slow parameter update in NMT structures. In LSTMs, the gradient vanishing [26] which occurs over time steps and over stacks in the vertical direction of the structures is resolved, respectively, by using memory cells and input feeding or multidimensional memories [15,27]. They are applied to the encoder and decoder, but the interface part is often a feedforward layer suffering from gradient vanishing. This vanishing limits the achievable translation quality in general and may restrict the learning of the correct composition.

Using a different type of cost function is an effective method to solve this problem. In our preliminary work [4], we evaluated the performance of applying a matching method for an English to French translation task with 1.5 million sentences and confirmed its improvement. We extended the work for more rigorous analysis to learn about how to control the hyper-parameters with respect to their impact on training.

3.2. Methods

The sentence representation matching proposed in this paper is composed of two ideas. First, this method use the output vector generated from the encoder model and passes it to the matching layer to derive the cost function. This idea reduces the distance from the cost function to an encoder model so that it can reduce the potential negative effects of gradient vanishing. Second, the cost is directly connected to the target word vectors, which is expected to induce an effective guide for the training of the encoder model.

To implement the two ideas as a single neural network added to existing NMTs, we introduce *sentence representation matching*, where we will call the sentence representation a concept implying the semantics of source or target sentences. An expected role of this approach is to guide NMTs not to train obviously wrong sentences in explicit direct mapping models.

The method illustrated in Figure 1 is the simplest, which will be extended in the following sections.

We newly propose the following three parts compared to NMT frameworks.

3.2.1. Vector Representation of Sentences

The proposed method matches two vector representations to evaluate the semantics of source and target sentences. The sentence representation of the source sentence is generated from the output vectors of an encoder model to induce a guiding effect to the model. The representation of the target sentence uses fixed-word vector sequences. The common

assumption in building the representations is to use fixed word vectors of an imported external dictionary or using one-hot representation.

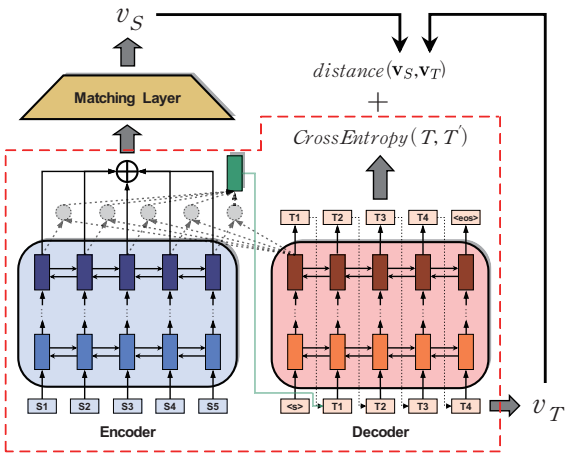

Figure 1. Sentence representation matching method plugged in to typical neural machine translation (red and dashed line: typical model).

The method used in this paper is defined according to the following equations, where the representations $\mathbf{r}_S$ of a source sentence S and $\mathbf{r}_T$ of a target sentence T are defined as

$$\mathbf{r}_S = \sum_{t=1}^{|S|} \mathbf{h}_t \quad (1)$$

$$\mathbf{r}_T = \sum_{t=1}^{|T|} \mathbf{w}_t \quad (2)$$

where $\mathbf{h}_t$ is the hidden vector generated from the top LSTM stacks at time step t in a NMT encoder and $\mathbf{w}_t$ is the word vector at time t in a decoder. This representation extraction is not neccessarily the addition of word or output vectors over times. It can be also easily extended to general encoder-decoder models. For example, in bidirectional models, $\mathbf{h}_t$ is replaced by $\mathbf{h}_t^f \| \mathbf{h}_t^b$. In bidirectional attention models, interface vectors are transformed vectors of $\mathbf{h}_t$ with the alignment model and target word, but we can still use $\mathbf{h}_t^f \| \mathbf{h}_t^b$.

3.2.2. Matching Layer

In matching representations, the biggest risk is the conflict of gradients to change the vector distribution of $\mathbf{h}_t$ between the cross-entropy and the distance, which generates bad local optima. If the scale of distance is dominated by the cross-entropy, the optima distribution will be similar to an original NMT, but otherwise, the negative phenomenon will happen. To reduce the negative effect of conflict, we added an additional linear combination layer for more flexible mapping, which can be extended to more general neural network layers.

$$\mathbf{v}_S = \mathbf{W}_r \mathbf{r}_S + \mathbf{b}_r \quad (3)$$

$$\mathbf{v}_T = \mathbf{r}_T \quad (4)$$

The matching layer is composed of parameters $\mathbf{W}_r$ and $\mathbf{b}_r$ for the representation matching. The generated source-side representation $\mathbf{v}_S$ is mapped to the target-side representation $\mathbf{v}_T$.

3.2.3. Cost Function

In sentence-level translation using representation matching, the underlying assumption is that the semantics of a source sentence and its translation are represented as the same vector in the space for representing general semantics. In the assumption, reducing the distance of the representations is exactly the same goal as cross-entropy in NMTs. For this reason, we believe that the cost function will not generate serious side-effects in training, but will induce some positive effects, such as providing a training guide and regularization, by providing information about correct translation in different perspectives. To use the information, we set a cost function as the following equation, given model parameter θ and training set D.

$$\mathcal{L}_{total} = \mathcal{L}_{cost} + \mathcal{L}_{distance}(\mathbf{v}_S, \mathbf{v}_T) \quad (5)$$

This method adds cost and distance without any scaling factors because the matching layer implicitly adapts its scale in updates. In early stages of the updates, the layer gives a large distance for all vectors by random initialization. but the long distance loss dominates updates and makes NMTs rapidly converge to a model to generate small distances over all vectors. Then, the impact of cross-entropy increases, and the model moves to the true optimal determined by the entropy. Therefore, if the optimal distance is sufficiently small, then this method will guide the training in early updates and preserve the true optima with respect to cross-entropy with the restriction of generating negative sentences.

3.3. Guiding Pressure

The newly introduced cost by matching sentence representations generates different gradients to cross-entropy, and then it pushes the model to move toward other directions. We call the gradients *guiding pressure* in this paper. This pressure is affected by many factors in matching methods because the gradients may disappear before changing the parameters of the encoder model by deep layers, large expression power of the matching layer, or loss propagated to the other layers. To understand the impact of the guiding pressure and to find the optimal configuration, we investigated potential factors of it and prepared the possible methods to control its strength.

3.3.1. Structure of Matching Layers

The depth of matching layers is a probable factor to affect the guidance strength. If the layers are deep, it causes the gradient vanishing problem, which weakens the impact of the guidance because the layers compose a feed-forward neural network. If the layers are too shallow, it restricts the expression power of the layers, and therefore, the strength of the guidance may be insufficiently increased.

In addition to the depth attribute, the responsible layer for the final cost calculation is also an attribute to affect the strength, because gradients to reduce the cost are equally contributed from source and target sides. To analyze the impact of the variation, we tested five structures, as shown in Figure 2 and Table 1.

Table 1. Definition of $\mathbf{v}_s$ and $\mathbf{v}_t$ for layer structures.

Structure	$\mathbf{v}_S$	$\mathbf{v}_T$
Source-side 1 layer	$\mathbf{W}_r\mathbf{r}_S + \mathbf{b}_r$ (Equation (3))	$\mathbf{r}_T$ (Equation (4))
Source-side 2 layers	$\mathbf{W}_{r_2}\text{sigm}(\mathbf{W}_{r_1}\mathbf{r}_S + \mathbf{b}_{r_1}) + \mathbf{b}_{r_2}$	$\mathbf{r}_T$
Target-side 1 layer	$\mathbf{r}_S$	$\mathbf{W}_r\mathbf{r}_T + \mathbf{b}_r$
Target-side 2 layers	$\mathbf{r}_S$	$\mathbf{W}_{r_2}\text{sigm}(\mathbf{W}_{r_1}\mathbf{r}_T + \mathbf{b}_{r_1}) + \mathbf{b}_{r_2}$
Both-sides 1 layer	$\mathbf{W}_r\mathbf{r}_S + \mathbf{b}_r$	$\mathbf{W}_r\mathbf{r}_T + \mathbf{b}_r$

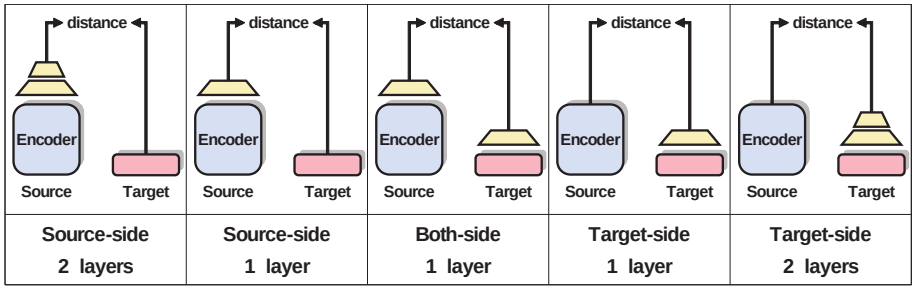

Figure 2. Layer structure for sentence representation matching.

3.3.2. Similarity Metrics

A metric to evaluate similarity between sentence representations also strongly affects the guiding pressure, because it determines points located at the same distance from the representation and gradients to move a matching representation without distance loss as well. We selected Hamming, Euclidean, and cosine distances for the distance method in (5), which have a rotated n-dimensional cube, n-dimensional sphere, and a line shape of the same distance region. We selected three types of widely used basic distance metrics, as shown in Table 2, which are defined by v_s and v_t of Equations (3) and (4) as follows.

$$\text{normalized Hamming distance} \quad : \quad \frac{||\mathbf{v}_s - \mathbf{v}_t||}{n} \tag{6}$$

$$\text{Euclidean distance} \quad : \quad ||\mathbf{v}_s - \mathbf{v}_t||_2 \tag{7}$$

$$\text{cosine similarity} \quad : \quad \frac{\mathbf{v}_s \mathbf{v}_t}{||\mathbf{v}_s|| ||\mathbf{v}_t||} \tag{8}$$

Table 2. Hyperparameters for grid search of the best sentence matching network in impact analysis.

Structure	Source-side 2 layers, Source-side 1 layer, Target-side 2 layers, Target-side 1 layer, Both-sides 1 layer
Metric	Hamming, Euclidean, cosine
Model capacity	Hidden nodes: $\{10, 50, 100, 250, 500\}$

3.3.3. Model Capacity

A clear attribute to decide the guide strength is model capacity, also understood as model complexity. If model complexity is low, the model can be too generalized to represent a complex relation for matching representation. Otherwise, the model easily finds the complex relation without propagating the pressure to the encoder model. An optimal model capacity is determined by involved sentence representations varying by given data, so that empirical investigation is required. From the preliminary results, we found that the two source-side layer allows for flexibly varying the model complexity with effective guiding, so we investigated various model capacities by changing the number of hidden nodes of the matching layer from an extremely low dimension to a sufficiently large dimension, not regularizing the network at all. Although the capacity depends on the source and target dictionary size, the size and computational complexity to train occupy a tiny portion of all.

4. Experiment Setting

In our experiments, we aim to investigate the impact of the representation matching with various parameter conditions because of its complex relation to regularization caused by many factors. Then, we evaluated its performance in translation of English from and to French, Spanish, and German.

4.1. Settings for Impact Analysis

To build a training set, we merged the Europarl parallel corpus and the Commoncrawl corpus released from WMT-14 (http://www.statmt.org/wmt14/, accessed on 18 January 2022). We applied tokenizing and lowercasing and limited the length of each sentence to a maximum value of 40 tokens through using scripts provided by a machine translation package, MOSES [28] (http://www.statmt.org/moses/, accessed on 18 January 2022). A starting and an ending symbol are attached to each source sentence. We used news-commentary-v8 and newstest-2013 sets released with the training set. Data statistics are shown in Table 3. For validation, we randomly selected 10% of the sentences of the training set.

Table 3. Data statistics for impact analysis (train: training set; test1: newstest-2013; test2: newscommentary-v8; validation set is a randomly selected 10% of the training set).

Type	Set	En	Fr	En	Es	En	DE	Unit
sentence	train	4.1		3.0		3.6		10^6
	test1	2623		2653		2741		10^0
	test2	117,861		138,408		151,139		10^0
token	train	83.8	92.2	62.9	65.6	76.4	72.6	10^6
	test1	48.0	53.8	48.0	51.7	52.6	51.4	10^3
	test2	2.5	2.9	3.0	3.3	3.3	3.4	10^6

We extracted word vectors from the training set using a language model implemented in word2vec [29] (https://code.google.com/archive/p/word2vec/, accessed on 18 January 2022) for all language pairs. The number of tokens in each dictionary is composed of the most frequently observed 40,000 tokens and their vector representations. In the training, validation, and test phases of a language pair, the same dictionary is imported.

We built a bidirectional model and passed the $\mathbf{h}$ and $\mathbf{c}$ from the forward to backward pass of the encoder. Then, the $\mathbf{h}_t$ of the forward pass and $\mathbf{h}_{|S|-t}$ are concatenated to derive $\mathbf{r}_s$. The used attention model is equal to [9], except for additionally passing $\mathbf{c}$ for the initialization of the decoder. The input word vectors are fed on to the second-shallowest LSTM stack of the encoder. To boost converging speed, we applied batch normalization through the weighted average of the original and normalized vectors. The weight is decayed by multiplying 0.8 at each epoch, which becomes almost 0 after 16 epochs. The representation matching is only applied to the training phase. Details of the model settings are shown in Table 4.

Table 4. Model settings for impact analysis (M: million, dim.: dimension).

LSTM stacks	4	parameter	
Cells per stacks	1000	encoder	3.05 M
Dim. of word	50	decoder	3.10 M
Dim. of attention	250	output	11 M
Batch size	128	interface	0.19 M

For the impact analysis of the guiding pressure, we extended this base structure with respect to the structure, model capacity, and distance metrics in English-French translation as shown in Table 3.

4.2. Setting for Robustness Analysis

In experiments to investigate the robustness of the model structure and embedding distribution, we reproduced a NMT open-source program (https://github.com/OpenNMT/OpenNMT-py, accessed on 18 January 2022) [30] and evaluated the performance of LSTM and the transformer with random, word2vec, and Bidirectional En-

coder Representations from Transformers (BERT) [31] embeddings for the French to English translation task. We used the re-sized Europarl corpus for training (https://www.statmt.org/europarl/v7/fr-en.tgz, accessed on 18 January 2022), common-test for validation (http://www.statmt.org/europarl/v1/common-test2.tgz, accessed on 18 January 2022), newstest (http://data.statmt.org/wmt17/translation-task/test.tgz, accessed on 18 January 2022) and news-commentary (http://matrix.statmt.org/test_sets/nc-test2007.tgz, accessed on 18 January 2022) for tests. Applied preprocessing methods are equal. The number of tokens are 50,000. Special symbols, including the start, end, unknown, and blank symbols, remained in the corpus. Data statistics for this robustness analysis are shown in Table 5.

Table 5. Data Statistics for robustness analysis in French to English translation (train: Europarl-v7, validation: commontest, test1: newstest-2014, test2: newscommentary-2007).

Corpus	Sentence (Fr-En)	Token (Fr)	Token (En)
Train	1,737,355	44,201,334	40,094,199
Valid	22,960	746,023	650,469
Test1	3,003	81,191	71,114
Test2	2,007	58,682	49,690

The tested embedding methods included random generation (RE), importing embedded vectors trained by word2vec (https://github.com/Andras7/word2vec-pytorch, accessed on 18 January 2022), implemented as open-source (WV), and importing embedded vectors of pre-trained multilingual BERT (BE). RE selects an element of an embedding vector from a uniform distribution in $[-0.1, 0.1]$, as shown in Table 6. WV is generated as in other experiments through learning a language model from the parallel training corpus with the same dictionary. BE is extracted from a pre-trained multilingual BERT [31] model (https://storage.googleapis.com/bert_models/2018_11_23/multi_cased_L-12_H-768_A-12.zip, accessed on 18 January 2022) consisting of an embedder and a number of encoders. RE, WV, and BE are fixed embedded vectors in every experiment.

Table 6. Statistics of embedding vectors for robustness analysis.

			Element Activation			
Embedding	**Unit**	**Language**	**Min**	**Max**	**Mean**	**Std**
WV	token	source	−0.8674	0.8900	−0.0003	0.0641
WV	token	target	−0.8819	0.8824	0.0018	0.0609
RE	token	source	−0.1000	0.1000	0.0000	0.0577
RE	token	target	−0.1000	0.1000	0.0000	0.0577
BE	word	source + target	−0.8911	0.4352	−0.0073	0.0463
BE	position	source + target	−0.8643	0.4100	0.0000	0.0150
BE	token	source + target	−0.3017	0.2326	0.0004	0.0178
			Euclidean Distance			
Embedding	**Unit**	**Language**	**Min**	**Max**	**Mean**	**Std**
WV	token	source	0.0248	5.9327	1.0426	1.0090
WV	token	target	0.0243	6.3314	0.8861	1.0553
RE	token	source	1.1810	1.4202	1.3061	0.0259
RE	token	target	1.1971	1.4040	1.3061	0.0257
BE	word	source + target	0.6753	1.8641	1.2880	0.1630
BE	position	source + target	0.3509	1.3160	0.4126	0.0465
BE	token	source + target	0.4846	0.5031	0.4939	0.0092

To compare the effects on widely used NMT architecture, we selected LSTM [10] with a unidirectional encoder and transformer [16]. In the transformer, the matching network

receives the final output vector of the encoder-transformer as its input and calculates its distance to the sum of word-embedding vectors of the decoder-transformer. Detailed model parameters are shown in Table 7. The setting of the matching layer is equal to the other experiments using source-side-2-layers, 250 hidden dimensions, and normalized hamming distance.

Table 7. Model settings for robustness analysis (M: million, voca.: vocabulary).

	LSTM		Transformer	
parameter	RE & WV	BE	RE & WV	BE
layers	4	4	6	2
number of heads	-	-	8	8
dimension of				
layer output	1000	1000	-	-
embedding	512	768	512	768
attention	1000	1000	1000	1000
model [16]	-	-	512	768
feed forward [16]	-	-	2048	2048
total parameters	168 M	373 M	120 M	303 M
batch size	64	32	64	32
voca. of encoder	50,002	119,547	50,002	119,547
voca. of decoder	50,004	119,547	50,004	119,547

5. Results and Discussions

To evaluate the impact on performance, we investigated token-level precision and BLEU scores [32]. In addition to these scores, we evaluated layer-wise statistics of a neural network to understand the impact on model complexity according to the following definition:

$$\mu_L = \frac{1}{|L|} \sum_{l \in L} \frac{||\mathrm{W}^{(l)}_{l_{in} \times l_{out}}||}{l_{in} l_{out}} \tag{9}$$

$$\sigma_L = \frac{1}{|L|} \sum_{l \in L} \sqrt{\frac{||\mathrm{W}^{(l)}_{l_{in} \times l_{out}}||_2^2}{l_{in} l_{out}} - \mu_l^2} \tag{10}$$

where L is a neural network layer and $\mathbf{W}^{(l)}_{l_{in} \times l_{out}}$ is its weight parameter matrix with l_{in} and l_{out} dimensions. The metrics μ_L and σ_L are the layer-wise mean and standard deviation of the network, and μ_l is the mean for the specific layer l. These two metrics are expected to show a degree of dispersion of hyperplanes represented by a layer in Cartesian and polar coordinates. Compared to the gradient, the change of the metrics is more focused on evaluating model complexity at each update step rather than the shape of movement of a model in optimization.

5.1. Impact Analysis

5.1.1. Structure

In Figure 3, the token-level precision results of each structure are shown. One-layer structures have a point in the graph because of the fixed dimension for the distance calculation. Two-layer structures are evaluated over varying hidden layer dimensions. The use of 1 or 2 layers showed no significant difference in source- and target-side structures. The both-sides structure showed less precision than the source-side structure.

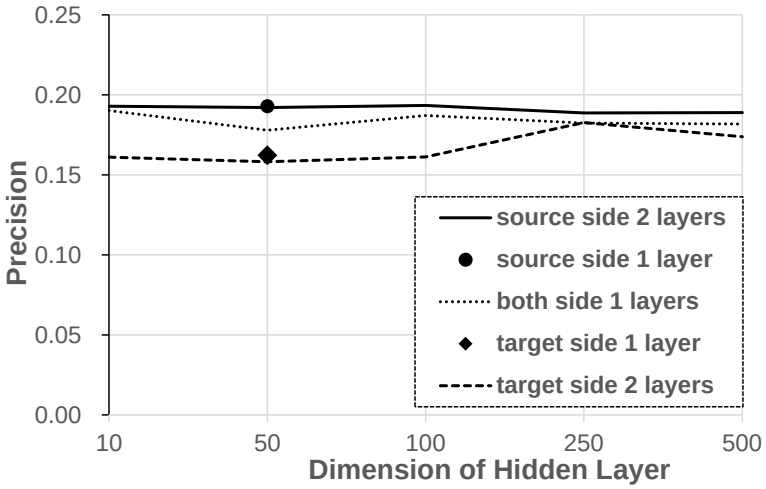

Figure 3. Token-level precision by dimensions of hidden layer for structure types.

Figure 4 shows the layer-wise mean and standard deviation of absolute weight parameters of encoder and decoder models in training. In the cases of the mean, the difference between structures is small because of the very large number of parameters. Thus, the small change may be induced by significantly large changes of some parameters. In the zoomed boxes of the subgraph (a), the mean values of the source-side layers were higher than the no-matching case and the target-side layers in the early epochs, although they were lower after sufficient training. In the subgraph (b), source-side layers maintained the mean value of the decoder models, but target-side layers showed significantly larger values. Subgraph (c) and (d) showed the similar superiority of source-side layers as the mean cases.

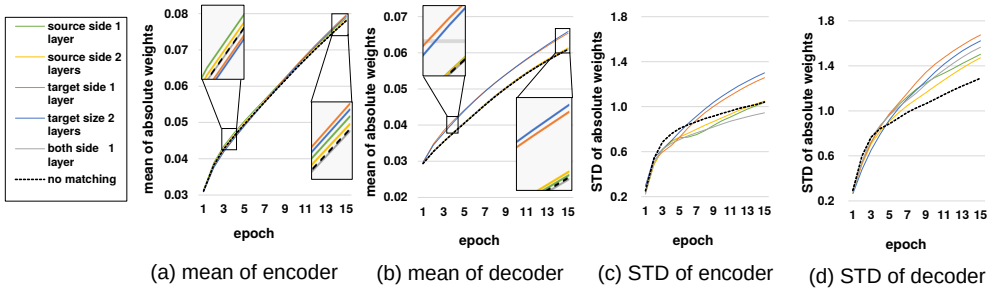

Figure 4. Layer-wise statistics μ_L and σ_L of weight parameters for a matching layer structure in training (cosine distance was used and the results were averaged over all hidden dimension settings).

In the overall results, source-side layers provided higher pressure at the early stage, but caused less conflict at the final stage of training compared to target-side layers. In the early stage, the gradient for the matching cost is split into the encoder and the matching layer in target-side layers, while source-side layers use the full gradient for training the encoder and increase training speed. In the final stage, the matching mechanism restricts the training of the correct output vector distribution of the encoder, so that the decoder models need to train more information and therefore increase the model complexity. In this stage, source-side layers are more flexible regarding changing the output distribution, because a neural network layer is a many-to-one mapping between the input and output. Any movement of the output distribution is regarded as the change of the input in the source-side layers, but it pushes the output vectors of the matching layer in target-side layers. Thus, target-side layers are more reluctant to change and restrict the encoder to move toward the correct distribution for minimizing the translation cost. In sum,

- Source-side layers have a higher impact on improving performance;
- Source-side layers have stronger pressure and less conflict.

5.1.2. Similarity Metrics

We evaluated the impact of similarity metrics in the source-side two layers case, as in Figure 5.

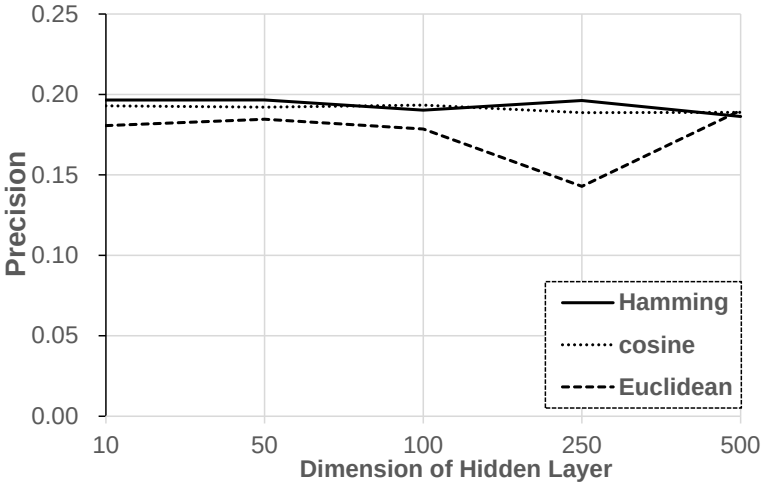

Figure 5. Token-level precision by dimensions of hidden layers for distance metrics.

In the results of Figure 5, the Hamming distance was slightly better than the cosine distance, and Euclidean distance showed much worse performance than the others. In Figure 6, the Hamming distance showed larger mean and STD than the cosine distance in all models and epochs. The case of Euclidean distance is excluded by extremely large values unfit to the scale to show the difference of other distance results. In the results, the impact of Euclidean distance on the pressure is the largest, but it seems to be stronger than the required pressure, as it restricts the performance. The Hamming and cosine distances are more stable, generating relatively weak pressure to preserve the model obtained by cross-entropy. The Hamming distance has stronger pressure than the cosine distance.

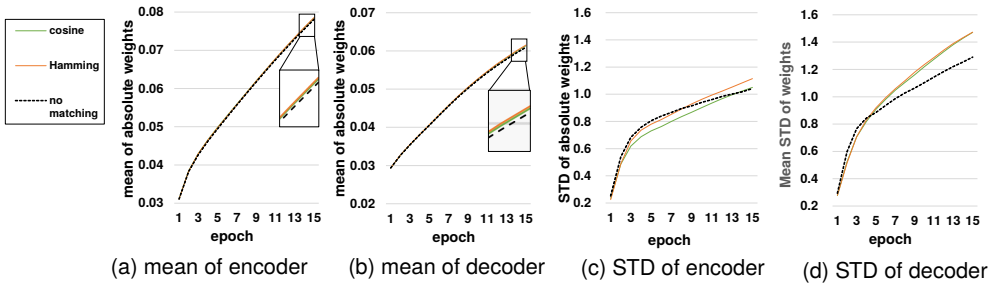

Figure 6. Layer-wise statistics μ_L and σ_L of weight parameters for distance types in training (cosine distance was used, and the results were averaged over all hidden dimension settings).

5.1.3. Model Capacity

The impact of the model capacity is shown in Figure 3, which includes the precision results of source, target, and both-sides 2 layers by changing the dimension of hidden layers.

Both-sides layers showed little change by increasing the dimension. The results of source-side layers slightly decreased the precision, but the target-side layers increased. In Figure 7, detailed results of the degree of dispersion are shown for all distance types with various dimensions. In the overall results, no simple correlation between hidden dimensions and the degrees of dispersion is found.

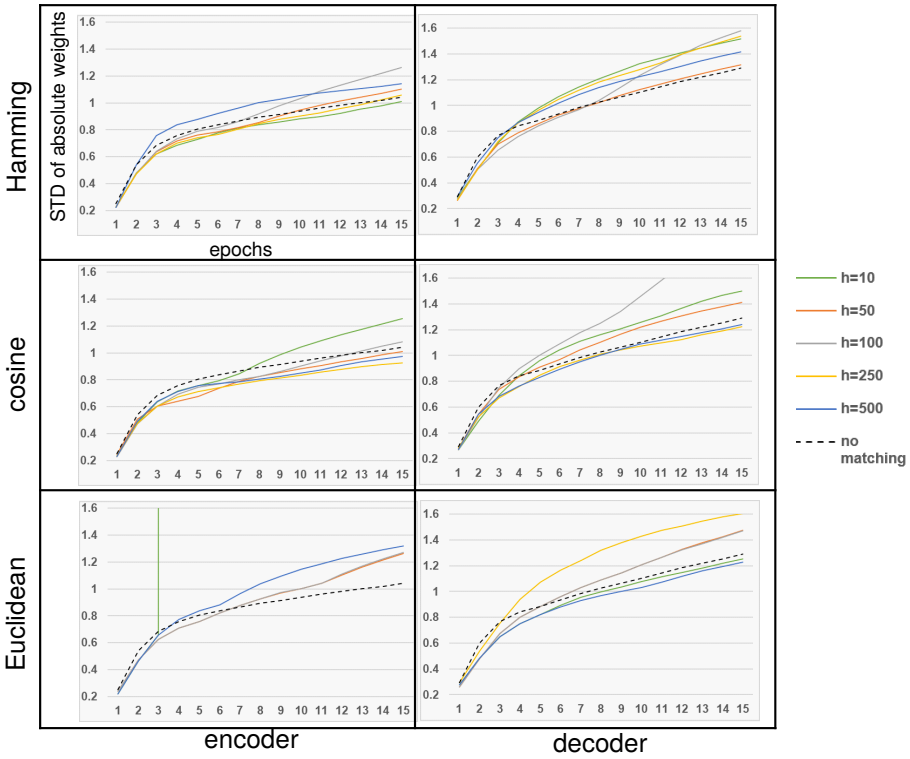

Figure 7. Layer-wise STD σ_L of 2 source-side layers by hidden dimension settings.

5.1.4. Performance of Best Matching Layer

In the French to English task used for the impact analysis, the grid search results are shown in the highlighted table and are drawn in Figure 8. The table shows the most achievable performance for validation and test sets, which are not affected by model selection based on validation performance. The best case in the table is that of the source-side 2 layer, 250 hidden dimension, and Hamming distance. We applied this setting for other translation tasks as an extension because of the large cost of the grid search.

From the impact investigation, we found the best settings to use included batch normalization, 2 source-side layers, 250 dimensions of hidden layers, and Hamming distance. With these settings, we performed translation tasks for three language pairs as shown in Table 8. In this table, we evaluated two different performances to show the robust result of the model selection process. One is the best performance in each data set evaluated at every epoch without a model selection process. The other is the best performance in each data set evaluated with the selected best validation model. The expected performance for all metrics and data sets using the matching method improves performance. In a more detailed view, the method improves the translation quality for the English to French, Spanish, and German pairs and the French to English pair. A notable point is that the precision in the training set is improved together when validation and test results are improved in the best validation model.

The result implies that the matching method is effective to improve the translation quality. It is more effective in improving the quality from relatively simpler language to more complex language in terms of tense and gender. A probable reason for the improvement is that the probability of generating the target sentences in those pairs is ambiguous in some cases, so the additional information given by the matching method clarifies the tokens to select. The performance improvement in the training and test sets of the best validation models is distinguished compared to the usual regularization methods, which increase the test accuracy by reducing the training accuracy.

structure	capacity	distance	valid		test1		test2	
			prec.	BLEU	prec.	BLEU	prec.	BLEU
s1	**50**	**cosine**	19.97	28.15	19.47	24.47	16.19	24.79
t1	**50**	**cosine**	17.84	22.00	16.57	17.12	13.66	16.90
s2	**10**	**cosine**	19.98	28.09	19.29	24.03	16.16	24.57
s2	**50**	**cosine**	19.88	28.16	19.29	24.51	16.12	24.57
s2	**100**	**cosine**	19.96	28.27	19.34	24.45	16.09	24.63
s2	**250**	**cosine**	19.98	28.21	19.10	24.69	16.02	24.74
s2	**500**	**cosine**	19.74	27.92	19.01	24.15	15.98	24.39
t2	**10**	**cosine**	17.82	22.10	16.69	17.23	13.70	17.03
t2	**50**	**cosine**	17.38	21.67	15.95	16.49	13.29	16.69
t2	**100**	**cosine**	17.40	21.44	16.26	16.45	13.42	16.38
t2	**250**	**cosine**	19.61	27.64	18.55	23.40	15.38	23.81
t2	**500**	**cosine**	18.89	26.37	17.65	21.99	14.88	22.19
b1	**10**	**cosine**	19.86	28.09	19.29	24.36	16.04	24.57
b1	**50**	**cosine**	19.95	28.08	19.37	24.70	16.26	24.55
b1	**100**	**cosine**	19.79	27.93	18.71	24.26	15.83	24.38
b1	**250**	**cosine**	19.35	27.67	18.24	23.86	15.26	23.83
b1	**500**	**cosine**	19.38	27.63	18.25	23.57	15.39	23.63
s2	**10**	**Hamming**	20.16	28.31	19.65	24.54	16.22	24.77
s2	**50**	**Hamming**	20.12	28.21	19.66	24.26	16.10	24.60
s2	**100**	**Hamming**	19.88	28.16	19.32	24.51	15.99	24.58
s2	**250**	**Hamming**	20.09	28.31	19.62	24.83	16.20	24.74
s2	**500**	**Hamming**	19.80	28.11	18.63	23.95	16.07	24.63
s2	**10**	**Euclidean**	18.57	26.95	18.06	23.12	14.94	23.36
s2	**50**	**Euclidean**	18.61	27.05	18.46	23.28	15.27	23.51
s2	**100**	**Euclidean**	18.46	26.78	17.92	23.20	14.77	23.00
s2	**250**	**Euclidean**	15.02	24.10	14.71	20.58	11.72	20.35
s2	**500**	**Euclidean**	19.69	27.46	19.01	23.51	15.80	23.66
no matching			19.25	27.65	18.45	23.74	15.57	24.08

Figure 8. Grid search results of best achievable performance for French to English translation (red: maximum, blue: minimum, white: mean, s1: source-side 1 layer, s2: source-side 2 layers, t1: target-side 1 layer, t2: target-side 2 layers, b1: both-sides 1 layer).

Table 8. Performance changes after sentence representation matching for various language pairs (δ: performance of models using the matching cost subtracted by original cost).

		Best Performance in Each Set					
		Valid		**Test1**		**Test2**	
Data	Model	Prec.	BLEU	Prec.	BLEU	Prec.	BLEU
	$E[\delta]$	0.32	0.33	0.23	0.32	0.05	0.30
En → Fr	NMT implementation of [9]	19.25	27.65	18.45	23.74	15.57	24.08
	NMT + matching	**20.09**	**28.31**	**19.62**	**24.83**	**16.20**	**24.74**
Fr → En	NMT implementation of [9]	28.36	27.97	22.96	22.40	21.14	23.03
	NMT + matching	28.25	**28.47**	22.88	**22.80**	**21.32**	**23.53**
En → Es	NMT implementation of [9]	26.55	31.14	20.48	22.43	24.21	30.55
	NMT + matching	**27.06**	**31.24**	20.30	22.30	23.13	30.28
Es → En	NMT implementation of [9]	30.41	32.37	23.24	22.39	25.09	30.31
	NMT + matching	29.98	32.32	22.84	**22.49**	**25.82**	**30.60**
En → De	NMT implementation of [9]	22.78	17.73	18.55	13.86	16.69	14.82
	NMT + matching	**24.32**	**19.22**	**20.53**	**15.17**	**17.92**	**16.31**
De → En	NMT implementation of [9]	23.33	23.59	20.05	18.31	18.44	20.31
	NMT + matching	22.88	22.86	18.93	17.44	17.03	19.43

Table 8. *Cont.*

		Performance of Best Validation Model				
		Train	Test1		Test2	
Data	Model	Prec.	Prec.	BLEU	Prec.	BLEU
	$E[\delta]$	0.99	0.25	0.72	0.27	0.69
En → Fr	NMT implementation of [9]	65.89	17.99	22.88	15.26	23.64
	NMT + matching	64.09	**19.62**	**24.41**	**16.20**	**24.74**
Fr → En	NMT implementation of [9]	63.22	22.56	21.68	17.98	21.94
	NMT + matching	**65.04**	22.48	**22.51**	17.94	**22.59**
En → Es	NMT implementation of [9]	63.21	19.66	20.96	21.27	28.41
	NMT + matching	**65.63**	**20.16**	**22.09**	**22.50**	**29.96**
Es → En	NMT implementation of [9]	65.42	22.23	21.47	22.42	28.62
	NMT + matching	63.83	20.18	20.24	20.39	27.24
En → De	NMT implementation of [9]	54.85	17.46	12.07	13.61	12.58
	NMT + matching	**61.24**	**20.53**	**14.87**	**17.00**	**15.81**
De → En	NMT implementation of [9]	60.75	19.88	17.59	17.18	19.68
	NMT + matching	59.46	18.31	16.84	15.28	18.68

5.2. Robustness Analysis

Table 9 shows the change in performance which occurred by applying the matching method. The bold scores indicate the cases that showed improvement. In the overall embedding and model settings, at least half of the test cases increased the scores. Precision and BLEU were improved up to 0.99 and 0.94. Transformer showed better performance than LSTM, but LSTM showed more cases that improved their performance. In the comparison of results between embedding methods, BE showed critically lower performance than RE and WV embedding.

In the observation, we could confirm that the matching method has more of a chance to cause positive effects in various models and embedding methods. The better performance of transformer compared to LSTM is consistently observed in current NMT literature. The benefit of the matching method is likely to be less positive in the transformer. This difference may be caused by different densities of information in the sentence representation generated by the encoder-transformer. The performance of the BE method is seriously worse than the others, but it is because of the big difference in the unknown word rate. The rate of commonly used tokens is a maximum of 9.36% in BE, while the others showed values as high as 48.32%, as shown in Table 10. The scale difference of input elements in embedded vectors was not a cause, as shown in Table 6.

Figure 9 shows three examples of robust analysis tests. Most translations are semantically almost correct translations in comparison to their reference sentence. However, the styles of models before and after applying matching are distinguished as red and blue-colored texts. Even if the semantic meaning is almost equivalent, the matching method more strongly follows the translation of the reference sentence. This is because the matching method has the role of restricting the translation to not generate a completely syntactic form in addition to the original translation models.

Table 9. Performance in robustness analysis (French to English translation, newstest: newstest-2014, nc-test:newscommentary-2007).

		Best Performance in Each Set					
		Common-Test		**Newstest**		**Nc-Test**	
Embedding	Networks	prec.	BLEU	prec.	BLEU	prec.	BLEU
Random Embedding (RE)	LSTM	5.72	10.96	14.08	21.30	16.73	27.29
	LSTM + matching	**5.82**	**11.10**	**14.25**	21.30	16.19	26.74
	Transformer	6.09	11.67	16.24	25.81	17.44	29.19
	Transformer + matching	6.09	11.61	**16.28**	24.96	**18.13**	**29.32**
Word2Vec (WV)	LSTM	5.77	11.12	14.55	21.75	16.67	26.57
	LSTM + matching	**5.95**	**11.45**	**14.73**	21.48	15.82	**26.78**
	Transformer	6.12	11.44	16.30	23.50	17.76	28.25
	Transformer + matching	6.10	11.33	16.07	**24.60**	**18.70**	**28.46**
BERT Embedding (BE)	LSTM	4.55	7.54	10.40	12.62	10.90	14.21
	LSTM + matching	**4.66**	**7.62**	**11.01**	**12.87**	**11.06**	**14.41**
	Transformer	2.86	2.49	5.26	2.64	6.19	3.54
	Transformer + matching	**3.30**	1.86	**6.25**	2.36	**6.97**	2.74

Table 10. Overlapping rate of sords and tokens between imported embedding dictionary and data sets for robustness analysis.

Dataset	**Sentence**	**RE & WV (50,000 Tokens)**				**BE (119,547 Tokens)**			
		Word		**Token**		**Word**		**Token**	
		Fr	**En**	**Fr**	**En**	**Fr**	**En**	**Fr**	**En**
Europarl	1,737,355	38.95	48.32	99.71	99.84	7.01	9.35	78.03	89.40
common-test	22,960	87.53	89.49	99.51	99.63	17.88	31.85	79.40	89.86
newstest	3003	81.80	82.39	95.92	96.08	30.44	45.33	74.87	83.07
nc-test	2007	92.17	92.87	98.67	98.65	31.89	48.34	75.26	84.05

5.3. Performance on State-of-the-Art Model

Table 11 shows the BLEU score on the BiBERT model with and without our matching method. We downloaded the IWSLT'14 *De* ↔ *En* data from Pytorch (https://github.com/pytorch/fairseq/blob/main/examples/translation/prepare-iwslt14.sh, accessed on 18 January 2022) and the code from BiBERT github link (https://github.com/fe1ixxu/BiBERT, accessed on 18 January 2022). Only the matching method was added, and all other progress of training were the same as baseline. The improvement of performance is observed in every condition, including the one-way, dual training, and fine-tuning conditions, except *En* → *De* with one-way training.

Table 11. Comparison of dual-directional and ordinary (one-way) translation models in [20] including stochastic layer selection ($K = 8$) with and without sentence representation mapping on IWSLT'14 *De* ↔ *En*.

Method	**De → En**		**En → De**	
	Baseline [20]	**+ Matching**	**Baseline [20]**	**+ Matching**
One-Way (vocab size = 12 K)	37.69	**38.13**	**30.00**	29.93
Dual-Directional Training	38.37	**38.42**	30.30	**30.50**
+ Fine-Tuning	38.61	**38.70**	30.45	**30.53**

case	sentence	BLEU
source No.1	**mais ces valeurs sont profondément enracinées en europe et elles doivent être déracinées de nouveau .**	
1-reference	**but these values are deeply ingrained in europe , and should be brought out again .**	
RE+LSTM	however , these values are deeply rooted in europe and must be restored .	20.82
RE+LSTM+matching	but these values are deeply rooted in europe and must be revived .	28.87
RE+transformer	however , these values are deeply rooted in europe and must be further excluded .	20.80
RE+transformer+matching	but these values are deeply rooted in europe and they have to be broken again .	30.86
WV+LSTM	however , these values are deeply rooted in europe and must be once again .	22.52
WV+LSTM+matching	but these values are deeply rooted in europe and they must be once again uprooted .	29.48
WV+transformer	however , these values are deeply rooted in europe and must be brought back .	22.52
WV+transformer+matching	but these values are deeply rooted in europe and they must be further uprooted .	28.94
source No.2	**tout semblait avoir suffisamment bien commencé .**	
1-reference	**everything seemed to have started well enough .**	
RE+LSTM	it seems to have got off to a good start .	19.04
RE+LSTM+matching	everything seemed well enough to start .	28.22
RE+transformer	everything seemed to have begun well enough .	50.00
RE+transformer+matching	everything seemed to have started well enough .	100.00
WV+LSTM	it all seemed quite well done .	22.72
WV+LSTM+matching	everything seemed to be too good .	24.93
WV+transformer	everything seemed to have started sufficiently well .	56.23
WV+transformer+matching	everything seemed to have started well enough .	100.00
source No.3	**taiwan n' a pas bénéficié de subventions étrangères ni d' accès privilégié à certains marchés .**	
1-reference	**taiwan did not receive foreign aid or preferential market access .**	
RE+LSTM	taiwan has not benefited from foreign subsidies or privileged access to certain markets .	26.69
RE+LSTM+matching	taiwan did not benefit from foreign subsidies or privileged access to certain markets .	19.02
RE+transformer	taiwan has not been granted foreign subsidies or privileged access to certain markets .	29.69
RE+transformer+matching	taiwan has not had foreign subsidies or privileged access to some markets .	29.91
WV+LSTM	taiwan has not received any foreign subsidies or privileged access to certain markets .	29.69
WV+LSTM+matching	taiwan has not received foreign subsidies or privileged access to certain markets .	29.91
WV+transformer	taiwan has not been receiving foreign subsidies or privileged access to certain markets .	29.69
WV+transformer+matching	taiwan did not receive foreign subsidies or privileged access to certain markets .	32.52

Figure 9. Examples of translation from French to English in robustness analysis.

6. Conclusions

In this paper, we raised the issue of inefficiency in training the encoder of NMTs implemented as a conditional language model. To relax the limit, we introduced sentence representation matching to force the representations of a source and its corresponding target sentence to be closely located by adding their distance to a loss function. The impact analysis showed that source-side layers are more effective in training with lower conflict, and Hamming distance has stronger pressure than cosine distance. In the grid search, a 2-layer source-side structure with 250 hidden dimensions and Hamming distance showed the best performance in French to English translation. When translating language pairs between English, Spanish and German, this setting slightly improved translation quality. In a more generalized environment using transformer and the various embedding method, importing constant vectors from explicit resources, the matching method was slightly but more likely to increase translation performance. Sentence representation matching has specific patterns with respect to structure, distance, and capacity, but the best setting was in somewhat intermediate states. For this reason, it requires grid search to apply this method for more general applications, and the best setting found in this paper can provide a good initial point to search.

7. Future Work

The simplest approach to control the guiding pressure is to use balancing parameters to learn the scale of the distance. The approach can effectively change the guiding pressure, but it is still under the limitation determined by the architectural factors discussed in this paper. Their combination will provide more fine control of hyperparameters.

Author Contributions: Conceptualization, H.J. and K.K.; methodology, H.J. and K.K.; software, H.J. and K.K.; validation, H.J. and K.K. formal analysis, H.J. and K.K.; H.J. and K.K.; resources, H.J. and K.K.; data curation, H.J. and K.K.; writing—original draft preparation, H.J., S.W. and K.K.; writing—review and editing, J.-H.S., S.-H.N., S.J. and S.W.; visualization, S.W.; supervision, K.K.; project administration, K.K.; funding acquisition, K.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by an Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (R7119-16-1001, Core technology development of the real-time simultaneous speech translation based on knowledge enhancement) and supported by a Global University Project(GUP) grant funded by the GIST in 2019.

Data Availability Statement: Not applicable.

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Ranzato, M.; Chopra, S.; Auli, M.; Zaremba, W. Sequence level training with recurrent neural networks. *arXiv* **2015**, arXiv:1511.06732.
2. Shen, S.; Cheng, Y.; He, Z.; He, W.; Wu, H.; Sun, M.; Liu, Y. Minimum risk training for neural machine translation. *arXiv* **2015**, arXiv:1512.02433.
3. Jung, S.; Lee, J.; Kim, J. Learning to Embed Semantic Correspondence for Natural Language Understanding. In Proceedings of the 22nd Conference on Computational Natural Language Learning, Brussels, Belgium, 31 October–1 November 2018; Association for Computational Linguistics: Brussels, Belgium, 2018; pp. 131–140. [CrossRef]
4. Kim, K.; Shin, J.H.; Na, S.H.; Jung, S. Concept Equalization to Guide Correct Training of Neural Machine Translation. In Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers), Taipei, Taiwan, 27 November–1 December 2017; Asian Federation of Natural Language Processing: Taipei, Taiwan, 2017; pp. 302–307.
5. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
6. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Montreal, QC, Canada, 2014; pp. 3104–3112.
7. Schuster, M.; Paliwal, K.K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **1997**, *45*, 2673–2681. [CrossRef]
8. Sundermeyer, M.; Alkhouli, T.; Wuebker, J.; Ney, H. Translation modeling with bidirectional recurrent neural networks. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 14–25.
9. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
10. Luong, M.T.; Pham, H.; Manning, C.D. Effective approaches to attention-based neural machine translation. *arXiv* **2015**, arXiv:1508.04025.
11. Luong, M.T.; Sutskever, I.; Le, Q.V.; Vinyals, O.; Zaremba, W. Addressing the rare word problem in neural machine translation. *arXiv* **2014**, arXiv:1410.8206.
12. He, D.; Xia, Y.; Qin, T.; Wang, L.; Yu, N.; Liu, T.Y.; Ma, W.Y. Dual learning for machine translation. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 820–828.
13. Ahmadnia, B.; Dorr, B.J. Augmenting neural machine translation through round-trip training approach. *Open Comput. Sci.* **2019**, *9*, 268–278. [CrossRef]
14. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 448–456.
15. 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.
16. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: San Francisco, CA, USA, 2017; pp. 5998–6008.
17. Liu, X.; Duh, K.; Liu, L.; Gao, J. Very deep transformers for neural machine translation. *arXiv* **2020**, arXiv:2008.07772.
18. 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.
19. Lin, Z.; Pan, X.; Wang, M.; Qiu, X.; Feng, J.; Zhou, H.; Li, L. Pre-training Multilingual Neural Machine Translation by Leveraging Alignment Information. *arXiv* **2020**, arXiv:2010.03142.
20. Xu, H.; Van Durme, B.; Murray, K. BERT, mBERT, or BiBERT? A Study on Contextualized Embeddings for Neural Machine Translation. *arXiv* **2021**, arXiv:2109.04588.
21. AP, S.C.; Lauly, S.; Larochelle, H.; Khapra, M.; Ravindran, B.; Raykar, V.C.; Saha, A. An autoencoder approach to learning bilingual word representations. In *Advances in Neural Information Processing Systems*; MIT Press: Montreal, QC, Canada, 2014; pp. 1853–1861.
22. Luong, T.; Pham, H.; Manning, C.D. Bilingual Word Representations with Monolingual Quality in Mind. In Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing, Denver, CO, USA, 5 June 2015; pp. 151–159.
23. Upadhyay, S.; Faruqui, M.; Dyer, C.; Roth, D. Cross-lingual models of word embeddings: An empirical comparison. *arXiv* **2016**, arXiv:1604.00425.
24. Zaremba, W.; Sutskever, I.; Vinyals, O. Recurrent neural network regularization. *arXiv* **2014**, arXiv:1409.2329.

25. Vinyals, O.; Fortunato, M.; Jaitly, N. Pointer networks. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Montreal, QC, Canada, 2015; pp. 2692–2700.
26. Bengio, Y.; Simard, P.; Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Netw.* **1994**, *5*, 157–166. [CrossRef] [PubMed]
27. Kalchbrenner, N.; Danihelka, I.; Graves, A. Grid long short-term memory. *arXiv* **2015**, arXiv:1507.01526.
28. Koehn, P.; Hoang, H.; Birch, A.; Callison-Burch, C.; Federico, M.; Bertoldi, N.; Cowan, B.; Shen, W.; Moran, C.; Zens, R.; et al. Moses: Open source toolkit for statistical machine translation. In Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, Prague, Czech Republic, 25–27 June 2007; Association for Computational Linguistics: Prague, Czech Republic, 2007; pp. 177–180.
29. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
30. Klein, G.; Kim, Y.; Deng, Y.; Senellart, J.; Rush, A. OpenNMT: Open-Source Toolkit for Neural Machine Translation. In Proceedings of the ACL 2017, System Demonstrations, Vancouver, BC, Canada, 30 July–4 August 2017; Association for Computational Linguistics: Vancouver, BC, Canada, 2017; pp. 67–72.
31. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
32. 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, Philadelphia, PA, USA, 7–12 July 2002; pp. 311–318.