Modeling Voting for System Combination in Machine Translation

Xuancheng Huang[1] , **Jiacheng Zhang**[1] , **Zhixing Tan**[1] , **Derek F. Wong**[2] , **Huanbo Luan**[1] , **Jingfang Xu**[3] , **Maosong Sun**[1] and **Yang Liu**[1,4,5*]
[1]Dept. of Comp. Sci. & Tech., BNRist Center, Institute for AI, Tsinghua University
[2]NLP^{2}CT Lab / Department of Computer and Information Science, University of Macau
[3]Sogou Inc.
[4]Beijing Advanced Innovation Center for Language Resources
[5]Beijing Academy of Artificial Intelligence

Abstract

System combination is an important technique for combining the hypotheses of different machine translation systems to improve translation performance. Although early statistical approaches to system combination have been proven effective in analyzing the consensus between hypotheses, they suffer from the error propagation problem due to the use of pipelines. While this problem has been alleviated by end-to-end training of multi-source sequence-to-sequence models recently, these neural models do not explicitly analyze the relations between hypotheses and fail to capture their agreement because the attention to a word in a hypothesis is calculated independently, ignoring the fact that the word might occur in multiple hypotheses. In this work, we propose an approach to modeling voting for system combination in machine translation. The basic idea is to enable words in hypotheses from different systems to vote on words that are representative and should get involved in the generation process. This can be done by quantifying the influence of each voter and its preference for each candidate. Our approach combines the advantages of statistical and neural methods since it can not only analyze the relations between hypotheses but also allow for end-to-end training. Experiments show that our approach is capable of better taking advantage of the consensus between hypotheses and achieves significant improvements over state-of-the-art baselines on Chinese-English and English-German machine translation tasks. [1]

1 Introduction

Machine translation (MT) is a challenging artificial intelligence task. Although many methods have been proposed for MT [Brown *et al.*, 1993; Koehn *et al.*, 2003; Bahdanau *et al.*, 2015; Vaswani *et al.*, 2017], they can only capture partial regularities of the translation process due to the complexity and diversity of natural languages. To address this problem, *system combination*, which aims to combine the hypotheses of multiple MT systems to obtain better translations, has been intensively studied [Rosti *et al.*, 2007b; He *et al.*, 2008; Zhou *et al.*, 2017] and widely used in MT evaluations [Barrault *et al.*, 2019]. Table 1 shows an example. Given an input German sentence (i.e., src), there are three erroneous but complementary hypotheses (i.e., hyp_1, hyp_2, and hyp_3) generated by three different MT systems. The goal of system combination is to combine them to produce a better translation (i.e., trg).

src	Ich hatte gestern einen Kuchen gegessen.
hyp_1	I ate a cake.
hyp_2	I eat a cakes yesterday.
hyp_3	I ate a fish yesterday.
trg	I ate a cake yesterday.

Table 1: Example of system combination in machine translation. Given a German sentence (i.e., src), there are three hypotheses (i.e., hyp_1, hyp_2, and hyp_3) generated by three different German-English MT systems. The goal of system combination is to combine these erroneous but complementary hypotheses to obtain a better English translation (i.e., trg).

Approaches to system combination can be roughly divided into two broad categories: *statistical* and *neural* approaches. Among statistical approaches combining hypotheses at different levels [Bangalore *et al.*, 2001; Matusov *et al.*, 2006; Rosti *et al.*, 2007a; Rosti *et al.*, 2007b; He *et al.*, 2008; Karakos *et al.*, 2008; Chen *et al.*, 2009; Feng *et al.*, 2009; Freitag *et al.*, 2014; Ma and McKeown, 2015], word-level combination approaches [Rosti *et al.*, 2007b; He *et al.*, 2008] prove to achieve the best translation performance. These approaches are capable of analyzing the relations between hypotheses and *voting* on the most probable word at each position by using a pipeline: choosing a backbone, aligning the words between hypotheses, building a confusion network, and generating the translation. Despite the benefit of explicit modeling of voting, the use of pipelines often results in the error propagation problem: errors made in early steps in the pipeline will be propagated to subsequent steps.

*Yang Liu is the corresponding author: liuyang2011@tsinghua.edu.cn.

[1]We release our source code at Github: https://github.com/THUNLP-MT/Voting4SC

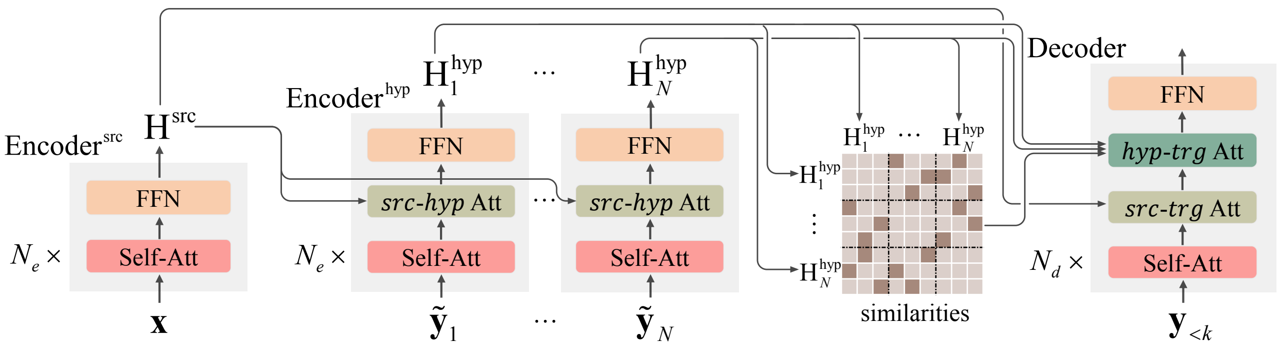


Figure 1: The architecture of our approach. Based on the multi-source sequence-to-sequence model, our approach introduces a voting mechanism to find the agreement between hypotheses. During voting, while a voter's influence depends on the source sentence $\mathbf{x}$ and the partial output $\mathbf{y}_{<k}$, its preference for a candidate depends on their word similarity calculated using the output of hypothesis encoders (i.e., $\mathbf{H}_n^{\text{hyp}}$). The result of voting is used to change the attention weights between hypotheses and output (i.e., *hyp-trg* Att) to encourage words receiving more votes to be more likely to be included in the output.

Recently, the error propagation problem has been alleviated by end-to-end training of neural combination methods [Zhou *et al.*, 2017], since system combination can be regarded as a multi-source sequence-to-sequence problem [Zoph and Knight, 2016].[2] One advantage of this line of work is that the importance of a word in a hypothesis can be quantified by its attention weight, without the need for explicit hypothesis alignment. However, a key limitation is that the attention weights between one hypothesis and the output are calculated independently, ignoring the fact that a candidate word might occur in multiple hypotheses. For example, in Table 1, as "ate" occurs in two hypotheses and "eat" occurs in one hypothesis, "ate" should be more likely to appear in the output. However, such connections between hypotheses are not taken into consideration in existing work.

In this work, we propose an approach to model voting for system combination in machine translation. The basic idea is to find the consensus between hypotheses by enabling words in hypotheses to vote on representative words. Our approach distinguishes between two types of words during voting: *voter* and *candidate*. For example, in Table 1, if "fish" in hyp_3 is chosen as a candidate, all words of hyp_1 and hyp_2 will serve as voters to decide whether "fish" should be included in the output. To do so, our approach quantifies the influence of each voter and its preference for each candidate using contexts and word similarities, respectively. By modifying attention weights, candidates receiving more votes are more likely to participate in the generation process. Our approach combines the merits of statistical and neural combination methods by both exploiting the relations between hypotheses and allowing for end-to-end training. Experiments show our approach achieves significant improvements over state-of-the-art statistical and neural combination methods on NIST and WMT benchmarks.

[2]By definition, system combination methods do not include model ensemble [Xiao *et al.*, 2013], which combines the predictions of multiple homogeneous models during decoding instead of combining the hypotheses of multiple heterogeneous MT systems after decoding.

2 Approach

Figure 1 shows the overall architecture of our model. The model starts with learning the representations of the source sentence and hypotheses (Section 2.1). Given the representation of hypotheses, our approach introduces voting by calculating the similarities between words in hypotheses and increasing the attention weights of similar words collectively (Section 2.2). Finally, the decoder takes the representations of contexts as input and encourages words on which most hypotheses agree to be more likely contribute to the generation of the output (Section 2.3).

2.1 The Encoders

As system combination takes multiple hypotheses from different MT systems as input, it is natural to cast system combination as a multi-source sequence-to-sequence problem [Zoph and Knight, 2016] as suggested by Zhou *et al.* [2017].

Formally, let $\mathbf{x}$ be a source sentence (i.e., src) and $\tilde{\mathbf{y}}_{1:N} = \tilde{\mathbf{y}}_1 \dots \tilde{\mathbf{y}}_N$ be N hypotheses generated by different MT systems, where $\tilde{\mathbf{y}}_n$ is the n-th hypothesis (i.e., hyp_n). We use $\tilde{y}_{n,j}$ to denote the j-th word of the n-th hypothesis. We use $\mathbf{y} = y_1 \dots y_K$ to denote the output (i.e., trg) with K words. Hence, the system combination model is given by

$$P(\mathbf{y}|\mathbf{x}, \tilde{\mathbf{y}}_{1:N}; \boldsymbol{\theta}) = \prod_{k=1}^{K} P(y_k|\mathbf{x}, \tilde{\mathbf{y}}_{1:N}, \mathbf{y}_{<k}; \boldsymbol{\theta}), \quad (1)$$

where y_k is the k-th target word, $\mathbf{y}_{<k} = y_1 \dots y_{k-1}$ is a partial output, and $\boldsymbol{\theta}$ is a set of model parameters.

To model the source sentence $\mathbf{x}$ and hypotheses $\tilde{\mathbf{y}}_{1:N}$, our model consists of $N+1$ encoders:

$$\mathbf{H}^{\text{src}} = \text{Encoder}^{\text{src}}(\mathbf{x}, \boldsymbol{\theta}), \quad (2)$$

$$\mathbf{H}_n^{\text{hyp}} = \text{Encoder}^{\text{hyp}}(\tilde{\mathbf{y}}_n, \boldsymbol{\theta}), 1 \le n \le N, \quad (3)$$

where $\text{Encoder}^{\text{src}}(\cdot)$ is the encoder for src, $\mathbf{H}^{\text{src}}$ is the representation of src, $\text{Encoder}^{\text{hyp}}(\cdot)$ is the encoder for each hypothesis, and $\mathbf{H}_n^{\text{hyp}}$ is the representation of hyp_n. Note that the parameters of the hypothesis encoder are shared by all hypotheses.

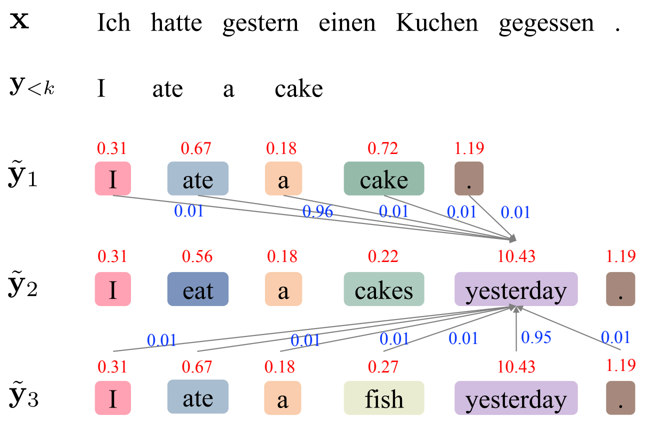


Figure 2: Voting in system combination. Similar words are highlighted in similar colors. "yesterday" in $\tilde{\mathbf{y}}_2$ is a candidate, which receives votes from voters in $\tilde{\mathbf{y}}_1$ and $\tilde{\mathbf{y}}_3$. The influence of each voter is a real-valued number highlighted in red and its preference for the candidate is highlighted in blue.

Although the multi-source sequence-to-sequence framework has shown the superiority of end-to-end training of neural combination methods over conventional statistical methods [Zhou *et al.*, 2017], the dependencies between hypotheses that are critical for finding the consensus are ignored because symbolic hypothesis alignment is not allowed in neural networks. Therefore, it is important to re-introduce voting into modern system combination methods while keeping the benefit of end-to-end training.

2.2 The Voting Mechanism

We first use an example to illustrate the basic idea of voting. Figure 2 shows a source sentence $\mathbf{x}$, a partial output $\mathbf{y}_{<k}$, and three hypotheses $\tilde{\mathbf{y}}_1$, $\tilde{\mathbf{y}}_2$, and $\tilde{\mathbf{y}}_3$. The question is which word in the hypotheses is more likely to be the fifth word of the output.

Our approach distinguishes between two kinds of words in hypotheses: *voter* and *candidate*. The voting mechanism allows each word (i.e., voter) to vote for other words (i.e., candidates) in other hypotheses. For example, the fifth word of $\tilde{\mathbf{y}}_2$ in Figure 2 (i.e., "yesterday") is a candidate and all words in $\tilde{\mathbf{y}}_1$ and $\tilde{\mathbf{y}}_3$ are voters. The voters decide whether the candidate should be included in the output by voting.

The voting process involves two basic problems:

1. *Influence*: how influential is the voter?
2. *Preference*: which candidate does the voter support?

The influence of each voter is quantified as a real-valued number (highlighted in red in Figure 2), which is actually the energy used in calculating attention weight:

$$e_{n,j} = f(\mathbf{x}, \mathbf{y}_{<k}, \tilde{y}_{n,j}, \boldsymbol{\theta}), \tag{4}$$

where $f(\cdot)$ is a function that calculates the energy, $\tilde{y}_{n,j}$ is the j-th word of the n-th hypothesis, and $e_{n,j}$ is its corresponding energy that reflects how likely it will be the next word. In Figure 2, "yesterday" receives the highest energy according to the source sentence and the partial output. Note that the energy of each word will change as the partial output changes. For example, the energy of "yesterday" will be decreased when the model is predicting the sixth word. This is much better than simply using number of occurrences that are independent of contexts as votes.

The preference of a voter for a candidate can also be measured as a real-valued number (highlighted in blue in Figure 2), which is actually the similarity between the voter and the candidate:

$$\text{sim}(\tilde{y}_{m,i}, \tilde{y}_{n,j}) = \frac{\exp(\mathbf{h}_{m,i}\mathbf{h}_{n,j}^\top)}{\sum_{i'=1}^{|\tilde{\mathbf{y}}_m|}\exp(\mathbf{h}_{m,i'}\mathbf{h}_{n,j}^\top)}, \tag{5}$$

where $\tilde{y}_{m,i}$ is a voter and $\mathbf{h}_{m,i}$ is its representation retrieved from $\mathbf{H}_m^{\text{hyp}}$. [3] Likewise, $\tilde{y}_{n,j}$ is a candidate and $\mathbf{h}_{n,j}$ is its representation. In Figure 2, "yesterday" is mainly supported by voters "yesterday" and "ate". Note that the similarities between the voters and a candidate are normalized at the hypothesis level to avoid the length bias: longer hypotheses tend to send out more votes.

It is easy to collect all votes by calculating a weighted sum of energies of voters. As a result, the extended energy of each candidate can defined as

$$\tilde{e}_{n,j} = e_{n,j} + \sum_{m=1 \wedge m \neq n}^{N} \sum_{i=1}^{|\tilde{\mathbf{y}}_m|} \text{sim}(\tilde{y}_{m,i}, \tilde{y}_{n,j}) \times e_{m,i}. \tag{6}$$

Clearly, the extended energy depends on both the influence (i.e., $e_{m,i}$) and the preference (i.e., $\text{sim}(\tilde{y}_{m,i}, \tilde{y}_{n,j})$).

Finally, the result of voting is used to change the attention weights between the hypotheses and output:

$$\alpha_{n,j} = \frac{\exp(\tilde{e}_{n,j})}{\sum_{n'=1}^{N}\sum_{j'=1}^{|\tilde{\mathbf{y}}_{n'}|}\exp(\tilde{e}_{n',j'})}. \tag{7}$$

In this way, candidates receiving strong support such as "yesterday" in Figure 2 will be more likely to appear in the output.

2.3 The Decoder

The decoder takes the representations of contexts as input and generates the output:

$$\mathbf{h}_k^{\text{trg}} = \text{Decoder}(\mathbf{y}_{<k}, \mathbf{H}^{\text{src}}, \mathbf{H}_{1:N}^{\text{hyp}}, \boldsymbol{\theta}), \tag{8}$$

$$P(y_k|\mathbf{x}, \tilde{\mathbf{y}}_{1:N}, \mathbf{y}_{<k}; \boldsymbol{\theta}) \propto g(\mathbf{h}_k^{\text{trg}}), \tag{9}$$

where $\text{Decoder}(\cdot)$ is the decoder, $\mathbf{h}_k^{\text{trg}}$ is the representation of the k-th target word y_k in trg, and $g(\cdot)$ is a function that calculates the generation probabilities. The difference between our $\text{Decoder}(\cdot)$ and the decoder described in Transformer [Vaswani *et al.*, 2017] is that our $\text{Decoder}(\cdot)$ contains additional hyp-trg attention layers, which replace the attention weight with $\alpha_{n,j}$ described in Eq.(7) (See Figure 1) .

When designing the decoder, we take advantage of two characteristics of system combination. First, words in hypotheses account for a large portion of those in the ground-truth output. Second, the hypotheses and output belong to the same language, making it convenient to detect and reduce word omission and repetition.

[3]We also tried other methods for calculating word similarity such as edit distance and the distance between word vectors but found that using the output of encoders works best because surrounding contexts are taken into consideration during representation learning.

Using both Restricted and Full Vocabularies

The first characteristic of system combination is that most words in hypotheses will appear in the ground-truth output. This is why conventional statistical combination methods [Rosti *et al.*, 2007b] only use a restricted vocabulary which contains words in hypotheses to constrain the search space. However, as there are still words in the ground-truth output falling outside the restricted vocabulary, modern neural combination methods often use the full vocabulary to ensure the coverage. Nevertheless, the observation that words in the hypotheses are more likely to be included in the output is not fully exploited.

As a result, we propose to use both restricted and full vocabularies to both take advantage of the observed regularity and ensure coverage. This can be done by calculating the output probabilities using restricted and full vocabularies separately and interpolating them using a gate:

$$\begin{aligned} & P(y_k|\mathbf{x}, \tilde{\mathbf{y}}_{1:N}, \mathbf{y}_{<k}; \boldsymbol{\theta}) \\ = \quad & \lambda_k \times P_r(y_k|\mathbf{x}, \tilde{\mathbf{y}}_{1:N}, \mathbf{y}_{<k}; \boldsymbol{\theta}) + \\ & (1-\lambda_k) \times P_f(y_k|\mathbf{x}, \tilde{\mathbf{y}}_{1:N}, \mathbf{y}_{<k}; \boldsymbol{\theta}), \end{aligned} \tag{10}$$

where $P_r(\cdot)$ is the probability calculated using the restricted vocabulary, $P_f(\cdot)$ is the probability calculated using the full vocabulary, and λ_k is a gate for predicting y_k. Note that λ_k obtained by $\mathbf{h}_k^{\text{trg}}$ after a simple linear transformation and Sigmoid activation function, which depends on the context and is not fixed during decoding.

Improving Coverage using Dynamic Weighting

Due to the lack of coverage vector that indicates whether a source word is translated or not [Koehn *et al.*, 2003], word omission and repetition are severe problems in neural MT systems that hinder translation performance [Tu *et al.*, 2016]. Fortunately, in system combination the hypotheses and output are in the same language, making it possible to detect and reduce word omission and repetition. For example, in Figure 2, as "cake" has already been included in the output, it should not appear in the output again. We propose to assign each word in hypotheses a weight, which changes dynamically during decoding: once a word is included in the output, its weight is decreased accordingly.

More precisely, we count the frequency of the words in hypotheses on average, which is represented by $\mathbf{c}^h \in \mathbb{R}^{|\mathcal{V}_f|}$. Meanwhile, we count the frequency of the words in $\mathbf{y}_{<k}$, which is denoted as $\mathbf{c}_k^y \in \mathbb{R}^{|\mathcal{V}_f|}$. Then, the probability distribution can be weighted by:

$$\mathbf{w}_k = \log_2\big(\max(\mathbf{c}^h - \mathbf{c}_k^y, 0) + 2\big), \tag{11}$$

$$\tilde{\mathbf{p}}_k \propto \mathbf{w}_k \odot \mathbf{p}_k, \tag{12}$$

where $\mathbf{w}_k \in \mathbb{R}^{|\mathcal{V}_f|}$ is a weight vector, $\tilde{\mathbf{p}}_k \in \mathbb{R}^{|\mathcal{V}_f|}$ is a weighted probability distribution for y_k, $\mathcal{V}_f$ is the full vocabulary, and $\odot$ is the point-wise multiplication. We keep the weight no smaller than 1.0 for each word and use a logarithmic function to deal with unexpected high frequency. At last, $\tilde{\mathbf{p}}_k$ is utilized by the beam-search algorithm to select top partial translations.

3 Experiments

3.1 Setup

Datasets

We evaluated our approach on Chinese-English (Zh-En) and English-German (En-De) translation tasks. For the Chinese-English task, the training set contains about 1.25M sentence pairs from LDC with 27.9M Chinese words and 34.5M English words. [4] We used the NIST 2006 dataset as the development set. The NIST 2002, 2003, 2004, 2005, and 2008 datasets were used as test sets. For the English-German task, the training set is the WMT 2014 training data with 4.5M sentence pairs, the validation set is newstest2013, and the test set is newstest2014. We used byte pair encoding (BPE) [Sennrich *et al.*, 2015] with 32K merges to segment words into sub-word units.

Evaluation Metrics

We report case-insensitive tokenized BLEU scores for Chinese-English and case-sensitive tokenized BLEU scores for English-German. For English-German, we apply compound splitting[5] similar to that of Vaswani *et al.* [2017]. We used the paired bootstrap resampling [Koehn, 2004] for statistical significance tests.

Single MT Systems

For the system combination task, we constructed all training datasets with outputs of three translation systems. We used the same systems for training, validation, and testing for both language pairs. Except for pretrained MT model, all single MT systems used the same training dataset as combination systems for training.

For the Chinese-English task, we used Transformer-base [Vaswani *et al.*, 2017] with left-to-right decoding (TRANS-L2R), Transformer-base with right-to-left decoding (TRANS-R2L) and non-autoregressive translation model (MASK-NAT) [Ghazvininejad *et al.*, 2019] as the three "black-box" translation systems. We used the training data simulation strategy [Zhou *et al.*, 2017] to alleviate the training-test bias.

For the English-German task, in order to demonstrate the effectiveness of our approach on top of the state-of-the-art results, we used FAIR's pretrained Transformer-big (TRANSFORMER$_{\text{big}}$-FB) [Ott *et al.*, 2018], Dynamic-Conv (DYNAMICCONV) [Wu *et al.*, 2019], and vanilla Transformer-big (TRANSFORMER$_{\text{big}}$) [Vaswani *et al.*, 2017] as individual MT systems.

Baselines

We compared our approach with the following two state-of-the-art statistical and neural combination methods:

1. JANE [Freitag *et al.*, 2014]: a statistical system combination method included in RWTH's open-source statistical machine translation toolkit. It was designed based on confusion networks [Rosti *et al.*, 2007b; He *et al.*, 2008]. We used JANE with its default setting.

[4]The training set includes LDC2002E18, LDC2003E07, LDC2003E14, part of LDC2004T07, LDC2004T08 and LDC2005T06.

[5]https://github.com/pytorch/fairseq/blob/master/scripts/compound_split_bleu.sh

Method	NIST02	NIST03	NIST04	NIST05	NIST08	All
Trans-R2L	45.11	44.67	46.66	46.08	36.90	44.26
Mask-NAT [Ghazvininejad *et al.*, 2019]	46.69	45.93	47.27	45.72	36.14	44.76
Trans-L2R [Vaswani *et al.*, 2017]	47.25	47.30	47.97	47.64	37.49	45.92
Jane [Freitag *et al.*, 2014]	47.75	47.88	48.90	48.83	38.66	46.78
Hier [Zhou *et al.*, 2017]	48.71	48.31	48.96	48.74	38.42	46.85
Ours	$\mathbf{49.30}^{\dagger\dagger\ddagger\ddagger**}$	$\mathbf{49.24}^{\dagger\dagger\ddagger\ddagger**}$	$\mathbf{49.65}^{\dagger\dagger\ddagger\ddagger**}$	$\mathbf{49.28}^{\dagger\dagger\ddagger**}$	$\mathbf{39.41}^{\dagger\dagger\ddagger\ddagger**}$	$\mathbf{47.69}^{\dagger\dagger\ddagger\ddagger**}$

Table 2: Results on the Chinese-English task. The evaluation metric is case-insensitive tokenized BLEU. "All" is the concatenation of all test sets. The translations of the top three single MT systems are the inputs of the bottom three system combination methods. "††": significantly better than "Transformer-L2R" ($p < 0.01$). "‡" and "‡‡": significantly better than "Jane" ($p < 0.05$ and $p < 0.01$). "**": significantly better than "Hier" ($p < 0.01$).

2. Hier [Zhou *et al.*, 2017]: a neural system combination method leveraging multi-source NMT framework [Zoph and Knight, 2016]. It was originally designed for the RNNsearch model [Bahdanau *et al.*, 2015]. To make a fair comparison, we adapted it to the Transformer model in our experiments since our approach is built on top of Transformer.

Hyper-parameter Setting

We used the same hyper-parameter setting for both baselines and our approach. The number of layers was set to 6 for both encoders and decoder. The hidden size was set to 512 and the filter size was set to 2,048. The number of individual attention heads was set to 8 for multi-head attention. We tied all three src, hyp, trg embeddings for the English-German task. The embeddings and softmax weights were tied for both language pairs. In training, we used Adam [Kingma and Ba, 2014] for optimization. Each mini-batch contains 19K tokens for the Chinese-English task and 25K tokens for the English-German task. We used the learning rate decay policy described by [Vaswani *et al.*, 2017]. In decoding, the beam size was set to 4 for both language pairs and the length penalty was set to 1.0 and 0.6 for Chinese-English and English-German, respectively. The other hyper-parameter settings were the same as the Transformer-base model [Vaswani *et al.*, 2017]. We used the development set to select the best model. We implemented our approach on top of the opensource toolkit THUMT[6].

3.2 Main Results

Table 2 shows the results on the Chinese-English task. We find that our method outperforms the best single system Trans-L2R, the statistical combination method Jane, and the neural combination method Hier. All the differences are statistically significant ($p < 0.01$). The superiority over Jane and Hier suggests that combining the merits of analyzing the dependencies between hypotheses and end-to-end training of neural networks helps to generate better translations.

Table 3 shows the results on the English-German task. Our approach also achieves significant improvements over the state-of-the-art results ($p < 0.01$). The gaps are relatively smaller than those on Chinese-English because the English-German task uses single reference while Chinese-English uses four references. Considering that Transformer$_{\text{big}}$-FB is nowadays acknowledged strongest single system result, Jane cannot improve the translation quality and Hier improves a little while our approach achieves significant improvements, indicating that voting mechanism and end-to-end training of neural networks is important especially when the translations of single systems already have high quality.

[6]https://github.com/THUNLP-MT/THUMT

Method	newstest2014
Transformer$_{\text{big}}$ [Vaswani *et al.*, 2017]	28.72
DynamicConv [Wu *et al.*, 2019]	29.74
Transformer$_{\text{big}}$-FB [Ott *et al.*, 2018]	29.76
Jane [Freitag *et al.*, 2014]	29.62
Hier [Zhou *et al.*, 2017]	29.95
Ours	$\mathbf{30.52}^{\dagger\dagger\ddagger\ddagger**}$

Table 3: Results on the English-German task. The evaluation metric is case-sensitive tokenized BLEU. The translations of the top three single MT systems are the inputs of the bottom three system combination methods. "††": significantly better than "Transformer$_{\text{big}}$-FB" ($p < 0.01$). "‡‡": significantly better than "Jane" ($p < 0.01$). "**": significantly better than "Hier" ($p < 0.01$).

Model	NIST06
Ours	**49.00**
− Dynamic Weighting	48.72
− Restricted Vocabulary	48.65
− Voting Mechanism	48.55

Table 4: Ablation study. "Dynamic Weighting" denotes improving coverage using dynamic weighting (see Section 2.3), "Restricted Vocabulary" denotes using both restricted and full vocabularies (see Section 2.3), and "Voting Mechanism" denotes the voting mechanism (see Section 2.2).

3.3 Ablation Study

Table 4 shows the results of ablation study. We find that the voting mechanism seems to play a critical role since removing it deteriorates the translation performance, which can be attributed to finding the consensus between hypotheses.

N	Single MT Systems					Training	Test	HIER	OURS	Δ
2	47.44	46.18	-	-	-	No	Yes	47.35	**47.76**†‡	+0.41
3	47.44	46.18	45.18	-	-	Yes	Yes	48.26	**49.00**††‡‡	+0.74
4	47.44	46.18	45.18	47.09	-	No	Yes	48.59	**49.56**††‡‡	+0.97
5	47.44	46.18	45.18	47.09	45.97	No	Yes	48.79	**49.80**††‡‡	+1.01

Table 5: Generalization ability evaluation. We report the results of single MT systems and system combination methods (i.e. HIER and OURS) on the Chinese-English task. N denotes the number of single MT systems. Note that both system combination methods were trained on the outputs of three single MT systems (i.e., $N = 3$) and tested on various number of single systems (i.e., $N = 2, 3, 4, 5$). "†" and "††": significantly better than the best system among inputs ($p < 0.05$ and $p < 0.01$). "‡" and "‡‡": significantly better than "HIER" ($p < 0.05$ and $p < 0.01$).

Both combining restricted and full vocabularies and dynamic weighting are shown to be beneficial for improving system combination but seem to have relatively smaller contributions than the voting mechanism.

3.4 Generalization Ability Evaluation

Table 5 shows the results of generalization ability evaluation. We are interested in evaluating whether a system combination method performs well when the number of single MT systems during testing is different from that during training.

In this experiment, we used the outputs of three single systems for training combination methods and tested them on two, three, four, and five single systems. [7] We find that our approach consistently improves over single systems while HIER underperforms the best single system when $N = 2$. In addition, the gap between our approach and HIER grows with the increase of N, indicating that the more hypotheses, the more effective the voting mechanism.

Specifically, we found that the number of single MT systems are more important than the averaged performance of single MT systems (See $N = 4$ and $N = 5$ in Table 5), indicating that the consensus of different hypotheses are more important than the quality of individual hypothesis. We shared the parameters of each hypothesis encoder $\mathrm{Encoder}^{\mathrm{hyp}}(\cdot)$ so that the system combination model has the ability to generalize to different number of single systems.

3.5 Effectiveness of Similarity Calculation

Table 6 shows the top-10 similar non-identical word pairs according to their similarities (See Eq.(5)). The similarity are averaged by every occurrences of the word pair on the Chinese-English development set. The higher the ranking, the higher the similarity. We only display word pairs that two words are non-identical because the similarity between two identical words are naturally high. However, we find that some non-identical word pairs have signally high similarities (e.g., the similarity between "kind" and "type", which is the highest ranked non-identical word pair, is ranked 37, higher than most of identical word pairs). It can be attributed to that different words might have similar meanings, indicating that our approach is able to catch the similar words in spite of their different morphologies.

[7]As JANE requires that the number of single systems during testing must be identical to that during training, it was not included in this experiment.

Rank	Word Pair	
37	kind	type
84	example	instance
96	made	makes
97	besides	except
151	greatest	broadest
156	make	makes
165	difficulties	difficulty
211	strategic	strategy
224	moreover	furthermore
228	7	7@@

Table 6: The top-10 similar non-identical word pairs. "Rank" refers to the rank among all word pairs, which is calculated according to their similarities in our approach (See Eq.(5)).

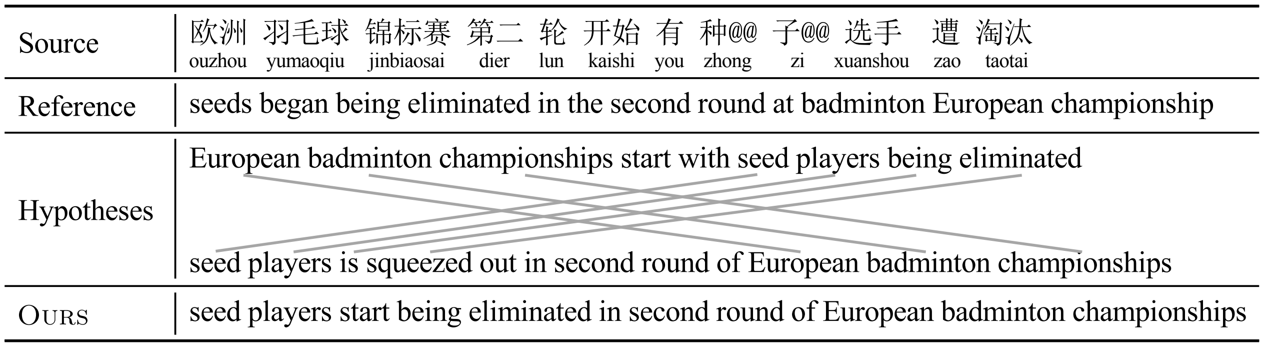
Figure 3 shows an example on using our approach to combine two single MT systems on the Chinese-English task. Word pairs with similarities higher than 0.12 are aligned with lines. We find that our approach is capable of identifying similar words such as "eliminated" and "squeezed" even if the two hypotheses have significantly different syntactic structures.

3.6 Matching Rate Evaluation

We attempt to evaluate whether our approach really leverage the consensus between hypotheses. Figure 4 shows the averaged matching rate between outputs and hypotheses in n-grams (n=1, 2, 3 and 4), respectively. The matching rate for individual n-gram between single hypothesis and output is the same as the matching rate in BLEU metric. The averaged one are averaged among different hypotheses, which reflects weather the output adopted the consensuses of hypotheses. As shown in this figure, JANE achieves highest averaged matching rate in 1-gram but as the interval becomes lager, its averaged matching rate decreases. Our method achieves highest averaged matching rate in n-grams ($n > 1$), which can be attributed to the voting mechanism and end-to-end training of neural architecture.

4 Related Work

System combination has been an active field in the past two decades and there are numerous valuable literatures address-



Source	欧洲 羽毛球 锦标赛 第二 轮 开始 有 种@@ 子@@ 选手 遭 淘汰 ouzhou yumaoqiu jinbiaosai dier lun kaishi you zhong zi xuanshou zao taotai
Reference	seeds began being eliminated in the second round at badminton European championship
Hypotheses	European badminton championships start with seed players being eliminated seed players is squeezed out in second round of European badminton championships
Ours	seed players start being eliminated in second round of European badminton championships

Figure 3: Case study. Word pairs with high similarities are aligned with lines. We find that our approach is able to identify similar words between hypotheses even if they differ in word orders significantly.

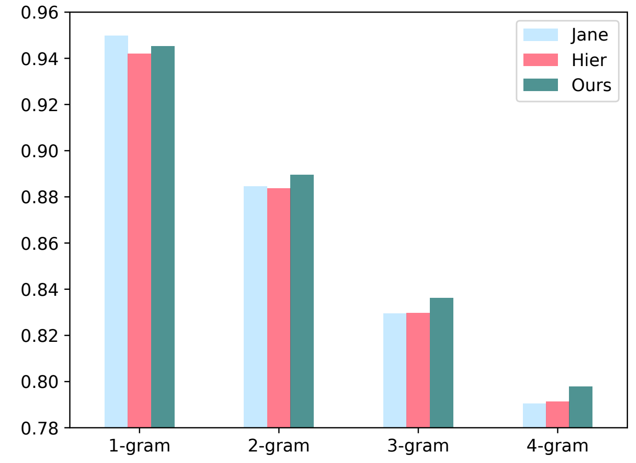


Figure 4: Averaged matching rate between outputs and hypotheses. We calculates matching rates of each hypothesis and output and averages them in n-grams (n=1, 2, 3 and 4), respectively, which is calculated on the Chinese-English development set.

ing it from different aspects [Bangalore *et al.*, 2001; Matusov *et al.*, 2006; Rosti *et al.*, 2007a; Rosti *et al.*, 2007b; Huang and Papineni, 2007; He *et al.*, 2008; Li and Zong, 2008; Karakos *et al.*, 2008; Feng *et al.*, 2009; Chen *et al.*, 2009; Du and Way, 2010; Heafield and Lavie, 2010; Ma and McKeown, 2012; Freitag *et al.*, 2014; Ma and McKeown, 2015; Freitag *et al.*, 2015; Zhu *et al.*, 2016; Zhou *et al.*, 2017; Barrault *et al.*, 2019].

Our idea of introducing voting into neural combination methods is inspired by classical confusion networks [Rosti *et al.*, 2007b; He *et al.*, 2008], which leverage word alignment between hypotheses to find groups of competing candidate words. The voting mechanism proposed in this work is significantly different from confusion networks in two aspects. First, our approach does not rely on a pipeline involving symbolic structures and thus facilitates end-to-end training of neural networks. This is important for alleviating the error propagation problem. Second, our approach leverages the contexts to compute votes dynamically. In other words, the influence and preference of a voter will change as the partial output changes.

The neural combination method proposed by Zhou *et al.* [2017] first shows the effectiveness of end-to-end training of multi-source sequence-to-sequence models in system combination. Along this direction, our work is also based on the same framework but focuses more on introducing voting into system combination. Our work shows that it is important to analyze the relations between hypotheses to find their consensus. As the attention weight between the hypothesis and output is calculated separately, we propose to increase the attention weights of a group of identical or similar words receiving high votes collectively.

Our work is also related to model ensemble [Xiao *et al.*, 2013] widely used in the deep learning community. The major difference is that model ensemble is a "white-box" method that aims to integrate predictions of multiple homogeneous models during inference while system combination is a "black-box" method that tries to combine hypotheses of multiple heterogeneous systems after inference.

5 Conclusion

We have presented a voting mechanism for system combination in machine translation. Our approach combines the advantages of statistical and neural methods by taking the relations between hypotheses into account and training models in an end-to-end manner. The voting mechanism allows words in hypotheses to vote on words that should be included in the output. Experiments show our approach achieves significant improvements over state-of-the-art baselines on Chinese-English and English-German translation tasks.

Acknowledgements

This work was supported by the National Key R&D Program of China (No.2017YFB0202204), National Natural Science Foundation of China (No.61925601, No.61761166008, No.61772302), Beijing Advanced Innovation Center for Language Resources (No.TYR17002), and the NExT++ project supported by the National Research Foundation, Prime Ministers Office, Singapore under its IRC@Singapore Funding Initiative. Derek F. Wong is supported by the National Natural Science Foundation of China (Grant No. 61672555), the Science and Technology Development Fund, Macau SAR (Grant Nos. 045/2017/AFJ, 0101/2019/A2).

References

[Bahdanau *et al.*, 2015] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

[Bangalore *et al.*, 2001] Bangalore Bangalore, Germán Bordel, and Giuseppe Riccardi. Computing consensus translation from multiple machine translation systems. *ASRU*, 2001.

[Barrault *et al.*, 2019] Loïc Barrault, Ondřej Bojar, Marta R Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, et al. Findings of the 2019 conference on machine translation (wmt19). In *WMT*, 2019.

[Brown *et al.*, 1993] Peter E. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *CL*, 1993.

[Chen *et al.*, 2009] Boxing Chen, Min Zhang, Haizhou Li, and AiTi Aw. A comparative study of hypothesis alignment and its improvement for machine translation system combination. In *ACL/IJCNLP*, 2009.

[Du and Way, 2010] Jinhua Du and Andy Way. Using terp to augment the system combination for smt. 2010.

[Feng *et al.*, 2009] Yang Feng, Yang Liu, Haitao Mi, Qun Liu, and Yajuan Lü. Lattice-based system combination for statistical machine translation. In *EMNLP*, 2009.

[Freitag *et al.*, 2014] Markus Freitag, Matthias Huck, and Hermann Ney. Jane: Open source machine translation system combination. In *EACL*, 2014.

[Freitag *et al.*, 2015] Markus Freitag, Jan-Thorsten Peter, Stephan Peitz, Minwei Feng, and Hermann Ney. Local system voting feature for machine translation system combination. In *WMT@EMNLP*, 2015.

[Ghazvininejad *et al.*, 2019] Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In *EMNLP/IJCNLP*, 2019.

[He *et al.*, 2008] Xiaodong He, Mei Yang, Jianfeng Gao, Patrick Nguyen, and Robert Moore. Indirect-HMM-based hypothesis alignment for combining outputs from machine translation systems. In *EMNLP*, 2008.

[Heafield and Lavie, 2010] Kenneth Heafield and Alon Lavie. Combining machine translation output with open source: The carnegie mellon multi-engine machine translation scheme. In *PBML*, 2010.

[Huang and Papineni, 2007] Fei Huang and Kishore Papineni. Hierarchical system combination for machine translation. In *EMNLP-CoNLL*, 2007.

[Karakos *et al.*, 2008] Damianos G. Karakos, Jason Eisner, Sanjeev Khudanpur, and Markus Dreyer. Machine translation system combination using itg-based alignments. In *ACL*, 2008.

[Kingma and Ba, 2014] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014.

[Koehn *et al.*, 2003] Philipp Koehn, Franz J. Och, and Daniel Marcu. Statistical phrase-based translation. In *NAACL*, 2003.

[Koehn, 2004] Philipp Koehn. Statistical significance tests for machine translation evaluation. In *EMNLP*, 2004.

[Li and Zong, 2008] Maoxi Li and Chengqing Zong. Word reordering alignment for combination of statistical machine translation systems. *ISCSLP*, 2008.

[Ma and McKeown, 2012] Wei-Yun Ma and Kathleen McKeown. Phrase-level system combination for machine translation based on target-to-target decoding. 2012.

[Ma and McKeown, 2015] Wei-Yun Ma and Kathleen McKeown. System combination for machine translation through paraphrasing. In *EMNLP*, 2015.

[Matusov *et al.*, 2006] Evgeny Matusov, Nicola Ueffing, and Hermann Ney. Computing consensus translation for multiple machine translation systems using enhanced hypothesis alignment. In *EACL*, 2006.

[Ott *et al.*, 2018] Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. Scaling neural machine translation. In *WMT*, 2018.

[Rosti *et al.*, 2007a] Antti-Veikko I. Rosti, Necip Fazil Ayan, Bing Xiang, Spyridon Matsoukas, Richard M. Schwartz, and Bonnie J. Dorr. Combining outputs from multiple machine translation systems. In *HLT-NAACL*, 2007.

[Rosti *et al.*, 2007b] Antti-Veikko I. Rosti, Spyridon Matsoukas, and Richard M. Schwartz. Improved word-level system combination for machine translation. In *ACL*, 2007.

[Sennrich *et al.*, 2015] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *ACL*, 2015.

[Tu *et al.*, 2016] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Modeling coverage for neural machine translation. In *ACL*, 2016.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[Wu *et al.*, 2019] Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. Pay less attention with lightweight and dynamic convolutions. *ICLR*, 2019.

[Xiao *et al.*, 2013] Tong Xiao, Jingbo Zhu, and T Liu. Bagging and boosting statistical machine translation systems. *AI*, 2013.

[Zhou *et al.*, 2017] Long Zhou, Wenpeng Hu, Jiajun Zhang, and Chengqing Zong. Neural system combination for machine translation. In *ACL*, 2017.

[Zhu *et al.*, 2016] Junguo Zhu, Muyun Yang, Sheng Li, and Tiejun Zhao. Sentence-level paraphrasing for machine translation system combination. In *ICYCSEE*, 2016.

[Zoph and Knight, 2016] Barret Zoph and Kevin Knight. Multi-source neural translation. In *HLT-NAACL*, 2016.