# TMU Japanese–English Neural Machine Translation System using Generative Adversarial Network for WAT 2018

**Yukio Matsumura** **Satoru Katsumata** **Mamoru Komachi**
Tokyo Metropolitan University
Tokyo, Japan

{matsumura-yukio, katsumata-satoru}@ed.tmu.ac.jp, komachi@tmu.ac.jp

## Abstract

This paper describes our neural machine translation (NMT) system. We implemented an attention-based recurrent neural network (RNN) encoder–decoder as a baseline. Additionally, we implemented a generative adversarial network (GAN) and reconstructor models in our NMT. We experimented with our NMT system on the shared tasks at the 5th Workshop on Asian Translation (WAT 2018). We participated in the scientific paper subtasks of the Japanese–English and English–Japanese translation tasks. The experimental results demonstrate that the ensemble of baseline systems achieved 25.85 and 36.14 points in Japanese–English and English–Japanese translations, respectively, in terms of BLEU scores. Furthermore, we found that GAN NMT can translate fluently.

## 1 Introduction

In recent years, neural machine translation (NMT) has been researched all over the world. Once the encoder–decoder NMT (Sutskever et al., 2014; Cho et al., 2014), which combines two recurrent neural networks (RNNs), was proposed, NMT gained huge popularity in the machine translation community.

However, the conventional encoder–decoder NMT works poorly on long sequences. Attention-based NMT (Bahdanau et al., 2015; Luong et al., 2015) can provide better prediction of output words by using the weights of each hidden state of the encoder as the context vector. It contributed to improvement of translation quality, especially in long sentences.

Transformer (Vaswani et al., 2017) is an extension of attention-based NMT; however, it is different from previous NMTs. They proposed a self-attention network and positional encoding. Thereby, NMT achieved high-quality translation without using RNN and convolutional neural network (CNN).

Nevertheless, NMT has several problems such as over-translation, wherein some words are translated repeatedly or unnecessary words are generated and under-translation, wherein some words remain mistakenly untranslated. Furthermore, an objective function of NMT is optimized by word unit; therefore, it cannot be guaranteed that the output of NMT is optimized as a sentence. This may become the cause of over- and under-translation.

In this paper, we describe the NMT system that was tested on the shared tasks at the 5th Workshop on Asian Translation (WAT 2018) (Nakazawa et al., 2018). We implemented an attention-based RNN encoder–decoder as a baseline. Furthermore, we implemented a generative adversarial network (GAN) and reconstructor models in our NMT.

GAN NMT comprises a generator and a discriminator. The discriminator should distinguish between true or generated sentences, whereas the generator aims to generate a sentence close to its correct translation, which the discriminator cannot distinguish. The goal of this adversarial training is to have the generator predict a target sentence close to its correct translation from given source sentence. Additionally, the objective function of this approach considers a term that is optimized by sentence unit. GAN is reported to improve translation quality (Yang et al., 2018).

Reconstructor NMT comprises an encoder–decoder and reconstructor. The reconstructor back-translates from hidden states of the decoder into the source sentence. On training, the NMT considers both: forward and back-translations. This approach can reduce over- and under-translation in forward translation because back-translation fails if there is a lack of information. The effect of this approach in English–Japanese translation is reported in (Matsumura et al., 2017).

We experimented with our NMT system for Japanese–English and English–Japanese scientific paper translation subtasks. The experimental results demonstrate that the ensemble of baseline systems achieved 25.85 and 36.14 points in Japanese–English and English–Japanese translations, respectively, in terms of BLEU (Papineni et al., 2002) scores. Furthermore, we found that GAN NMT can translate fluently in English-Japanese pairwise evaluation.

## 2 Attention-based NMT

In this section, we describe our baseline NMT system[1]. This system is based on the attention-based NMT (Luong et al., 2015). We adopted a bi-directional long short-term memory (LSTM) as the encoder and a unidirectional LSTM as the decoder.

### 2.1 Encoder

The source sentence is input as a sequence of one-hot word vectors: ($\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{|\boldsymbol{X}|}]$), where $|\boldsymbol{X}|$ is the length of the source sentence.

At each time step $i$, the source word embedding vector: $\boldsymbol{e}_i^s$ is computed by the following equation.

$$\boldsymbol{e}_i^s = \tanh(\boldsymbol{W}_x \boldsymbol{x}_i) \tag{1}$$

where $\boldsymbol{W}_x \in \mathbb{R}^{q \times v_s}$ is a weight matrix, $q$ is the dimension of the word embeddings, and $v_s$ is the size of the source vocabulary.

The hidden state $\bar{\boldsymbol{h}}_i$ of the encoder is computed by the following equation:

$$\bar{\boldsymbol{h}}_i = \overrightarrow{\boldsymbol{h}_i}^{(L)} + \overleftarrow{\boldsymbol{h}_i}^{(L)} \tag{2}$$

where $L$ is the number of layers. Here, the forward state $\overrightarrow{\boldsymbol{h}_i}^{(l)}$ and the backward state $\overleftarrow{\boldsymbol{h}_i}^{(l)}$ are computed by

$$\overrightarrow{\boldsymbol{h}_i}^{(l)} = \mathrm{LSTM}(\overrightarrow{\boldsymbol{h}_i}^{(l-1)}, \overrightarrow{\boldsymbol{h}_{i-1}}^{(l)}) \tag{3}$$

and

$$\overleftarrow{\boldsymbol{h}_i}^{(l)} = \mathrm{LSTM}(\overleftarrow{\boldsymbol{h}_i}^{(l-1)}, \overleftarrow{\boldsymbol{h}_{i+1}}^{(l)}) \tag{4}$$

where $l$ is the layer number. Note that $\overrightarrow{\boldsymbol{h}_i}^{(0)}$ and $\overleftarrow{\boldsymbol{h}_i}^{(0)}$ are regarded as $\boldsymbol{e}_i^s$.

### 2.2 Decoder

Similar to the source sentence, the target sentence is input as a sequence of one-hot word vectors: ($\boldsymbol{Y} = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_{|\boldsymbol{Y}|}]$), where $|\boldsymbol{Y}|$ is the length of the target sentence.

At each time step $j$, the hidden state $\boldsymbol{h}_j^{(l)}$ of each layer of the decoder is represented by the following equation.

$$\boldsymbol{h}_j^{(l)} = \mathrm{LSTM}(\boldsymbol{h}_j^{(l-1)}, \boldsymbol{h}_{j-1}^{(l)}) \tag{5}$$

Note that $\boldsymbol{h}_j^{(0)}$ is regarded as the concatenation of the target word embedding vector $\boldsymbol{e}_{j-1}^t$ and the attentional hidden state $\tilde{\boldsymbol{h}}_{j-1}$ at the previous time step: $[\boldsymbol{e}_{j-1}^t : \tilde{\boldsymbol{h}}_{j-1}]$. In this system, the first hidden state $\boldsymbol{h}_1{}^{(l)}$ of each layer is initialized by the hidden state of the encoder as follows:

$$\boldsymbol{h}_1{}^{(l)} = \overrightarrow{\boldsymbol{h}_{|\boldsymbol{X}|}}^{(l)} + \overleftarrow{\boldsymbol{h}_1}^{(l)}. \tag{6}$$

The target word embedding vector $\boldsymbol{e}_j^t$ is computed as:

$$\boldsymbol{e}_j^t = \tanh(\boldsymbol{W}_y \boldsymbol{y}_j) \tag{7}$$

where $\boldsymbol{W}_y \in \mathbb{R}^{q \times v_t}$ is a weight matrix and $v_t$ is the target vocabulary size. The attentional hidden state $\tilde{\boldsymbol{h}}_j$ is represented as:

$$\tilde{\boldsymbol{h}}_j = \tanh(\boldsymbol{W}_a[\boldsymbol{h}_j^{(L)} : \boldsymbol{c}_j] + \boldsymbol{b}_a) \tag{8}$$

where $\boldsymbol{W}_a \in \mathbb{R}^{r \times 2r}$ is a weight matrix, $\boldsymbol{b}_a \in \mathbb{R}^r$ is a bias vector, and $r$ is the number of hidden units.

The context vector $\boldsymbol{c}_j$ is a weighted sum of each hidden state $\bar{\boldsymbol{h}}_i$ of the encoder. It is represented as:

$$\boldsymbol{c}_j = \sum_{i=1}^{|\boldsymbol{X}|} \alpha_{ij} \bar{\boldsymbol{h}}_i. \tag{9}$$

Its weight $\alpha_{ij}$ is a normalized probability distribution, which is computed using a dot product of hidden states as follows:

$$\alpha_{ij} = \frac{\exp(\bar{\boldsymbol{h}}_i^{\mathrm{T}} \boldsymbol{h}_j^{(L)})}{\sum_{k=1}^{|\boldsymbol{X}|} \exp(\bar{\boldsymbol{h}}_k^{\mathrm{T}} \boldsymbol{h}_j^{(L)})}. \tag{10}$$

[1] https://github.com/yukio326/nmt-chainer

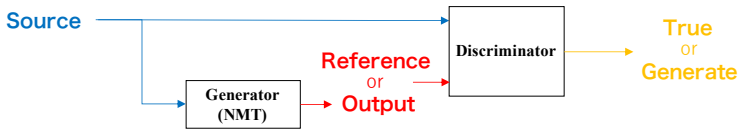


Figure 1: Overview of GAN NMT.

The conditional probability of the output word $\hat{\boldsymbol{y}}_j$ is computed as:

$$\boldsymbol{p}(\hat{\boldsymbol{y}}_j|\boldsymbol{Y}_{<j}, \boldsymbol{X}) = \text{softmax}(\boldsymbol{W}_p\tilde{\boldsymbol{h}}_j + \boldsymbol{b}_p) \quad (11)$$

where $\boldsymbol{W}_p \in \mathbb{R}^{v_t \times r}$ is a weight matrix and $\boldsymbol{b}_p \in \mathbb{R}^{v_t}$ is a bias vector.

### 2.3 Training

The objective function of this system is

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{D}\sum_{d=1}^{D}\sum_{j=1}^{|\boldsymbol{Y}|}\log \boldsymbol{p}(\boldsymbol{y}_j^{(d)}|\boldsymbol{Y}_{<j}^{(d)}, \boldsymbol{X}^{(d)}, \boldsymbol{\theta}) \quad (12)$$

where $D$ denotes the number of data and $\boldsymbol{\theta}$ denotes the model parameters. The model parameters in word embedding are pretrained using GloVe (Pennington et al., 2014). All other model parameters are randomly initialized.

### 2.4 Testing

To achieve better predictions, we adopted beam search and ensemble decoding. In beam search, the system retains hypotheses of beam size $n$ at each time step. During the subsequent time step, for each hypothesis, it computes $n$ hypotheses; further, it retains $n$ hypotheses out of the total $n^2$ hypotheses. In ensemble decoding, the conditional probability of the output word $\hat{\boldsymbol{y}}_j$ is the average of each model's score $\boldsymbol{p}^{(m)}$. It is computed by

$$\boldsymbol{p}(\hat{\boldsymbol{y}}_j|\boldsymbol{Y}_{<j}, \boldsymbol{X}) = \frac{1}{M}\sum_{m=1}^{M}\boldsymbol{p}^{(m)}(\hat{\boldsymbol{y}}_j|\boldsymbol{Y}_{<j}, \boldsymbol{X}) \quad (13)$$

where $M$ denotes the number of models. They reduce the risk of predicting wrong words.

## 3 GAN NMT

Herein, we describe GAN NMT[2] based on Yang et al. (2018). It comprises two networks: a generator which generates a target sentence, and a discriminator which distinguishes a generated sentence from its true translation as shown in Figure 1.

[2]https://github.com/yukio326/GAN-NMT

### 3.1 Generator

The generator attempts to generate a target sentence close to its correct translation from a given source sentence. We use the attention-based NMT described in Section 2 as the generator network.

### 3.2 Discriminator

The discriminator predicts whether the target sentence is true or generated by the given source and target sentences. At each time step $i$, the hidden state $\boldsymbol{f}_i^s$ corresponding to the source embedding vector $\boldsymbol{e}_i^s$ in Equation 1 is represented as:

$$\boldsymbol{f}_i^s = \overrightarrow{\boldsymbol{f}_i^s}(L) + \overleftarrow{\boldsymbol{f}_i^s}(L). \quad (14)$$

Here, the forward state $\overrightarrow{\boldsymbol{f}_i^s}(l)$ and the backward state $\overleftarrow{\boldsymbol{f}_i^s}(l)$ are computed by

$$\overrightarrow{\boldsymbol{f}_i^s}(l) = \text{LSTM}(\overrightarrow{\boldsymbol{f}_i^s}(l-1), \overrightarrow{\boldsymbol{f}_{i-1}^s}(l)) \quad (15)$$

and

$$\overleftarrow{\boldsymbol{f}_i^s}(l) = \text{LSTM}(\overleftarrow{\boldsymbol{f}_i^s}(l-1), \overleftarrow{\boldsymbol{f}_{i+1}^s}(l)). \quad (16)$$

Note that $\overrightarrow{\boldsymbol{f}_i^s}(0)$ and $\overleftarrow{\boldsymbol{f}_i^s}(0)$ are regarded as $\boldsymbol{e}_i^s$. The sentence vector of source sentence $\bar{\boldsymbol{f}}^s$ is computed by

$$\bar{\boldsymbol{f}}^s = \text{average}\left(\left[\boldsymbol{f}_1^s, \boldsymbol{f}_2^s, \cdots, \boldsymbol{f}_{|\boldsymbol{X}|}^s\right]\right). \quad (17)$$

Similarly, at each time step $j$, the hidden state $\boldsymbol{f}_j^s$ corresponding to the target embedding vector $\boldsymbol{e}_j^t$ in Equation 7 is represented as

$$\boldsymbol{f}_j^t = \overrightarrow{\boldsymbol{f}_j^t}(L) + \overleftarrow{\boldsymbol{f}_j^t}(L). \quad (18)$$

Here, the forward state $\overrightarrow{\boldsymbol{f}_j^t}(l)$ and the backward state $\overleftarrow{\boldsymbol{f}_j^t}(l)$ are computed by

$$\overrightarrow{\boldsymbol{f}_j^t}(l) = \text{LSTM}(\overrightarrow{\boldsymbol{f}_j^t}(l-1), \overrightarrow{\boldsymbol{f}_{j-1}^t}(l)) \quad (19)$$

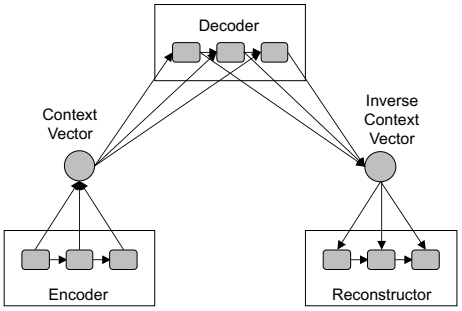


Figure 2: Overview of reconstructor NMT.

and

$$\overleftarrow{\boldsymbol{f}}_j^t(l) = \mathrm{LSTM}(\overleftarrow{\boldsymbol{f}}_j^t(l-1), \overleftarrow{\boldsymbol{f}_{j+1}^t}(l)). \quad (20)$$

Note that $\overrightarrow{\boldsymbol{f}}_j^t(0)$ and $\overleftarrow{\boldsymbol{f}}_j^t(0)$ are regarded as $\boldsymbol{e}_j^t$. The sentence vector of target sentence $\bar{\boldsymbol{f}}^t$ is computed by

$$\bar{\boldsymbol{f}}^t = \mathrm{average}\left(\left[\boldsymbol{f}_1^t, \boldsymbol{f}_2^t, \cdots, \boldsymbol{f}_{|\boldsymbol{Y}|}^t\right]\right). \quad (21)$$

Finally, the probability that the target sentence is true is predicted by the dot product of the source and target sentence vectors as follows:

$$p(\boldsymbol{X}, \boldsymbol{Y}) = \mathrm{sigmoid}(\bar{\boldsymbol{f}}^s \cdot \bar{\boldsymbol{f}}^t). \quad (22)$$

### 3.3 Training

GAN must be trained adversarially. The discriminator should distinguish between true or generated sentences, whereas the generator aims to generate a sentence close to its correct translation, which the discriminator cannot distinguish. Alternatively, the objective function of the generator differs from that of the discriminator.

The objective functions of the generator and discriminator networks are defined by the following:

**Generator**

$$\mathcal{L}_G(\boldsymbol{\theta}, \boldsymbol{\gamma}) = \frac{1}{D}\sum_{d=1}^{D}\Bigg\{\sum_{j=1}^{|\boldsymbol{Y}|}\log \boldsymbol{p}(\boldsymbol{y}_j^{(d)}|\boldsymbol{Y}_{<j}^{(d)}, \boldsymbol{X}^{(d)}, \boldsymbol{\theta}) + \log p(\boldsymbol{X}^{(d)}, \hat{\boldsymbol{Y}}^{(d)}|\boldsymbol{\gamma})\Bigg\}. \quad (23)$$

| | |
|---|---|
| train | 974,198 |
| dev | 1,790 |
| test | 1,812 |

Table 1: Number of Japanese–English parallel sentences.

**Discriminator**

$$\mathcal{L}_D(\boldsymbol{\gamma}) = \frac{1}{D}\sum_{d=1}^{D}\Bigg\{\log p(\boldsymbol{X}^{(d)}, \boldsymbol{Y}^{(d)}|\boldsymbol{\gamma}) + \log\Big\{1 - p(\boldsymbol{X}^{(d)}, \hat{\boldsymbol{Y}}^{(d)}|\boldsymbol{\gamma})\Big\}\Bigg\}. \quad (24)$$

where $\boldsymbol{\gamma}$ is the model parameters in the discriminator. In the objective function of generator, the second term considers the sentence unit information. We applied pre-training to both generator and discriminator using the baseline.

## 4 Reconstructor NMT

Next, we describe reconstructor NMT[3] based on Tu et al. (2017) as shown in Figure 2. It comprises two components: encoder–decoder and reconstructor, which back-translates from hidden states of the decoder into the source sentence. On training, the NMT considers both forward and back-translations. This approach can reduce over- and under-translation in forward translation because back-translation fails if there is a lack of information.

### 4.1 Encoder–Decoder

We use the attention-based NMT described in Section 2 as the encoder–decoder network. Difference from Matsumura et al. (2017) is an encoder–decoder network. Their encoder–decoder network is based on Bahdanau et al. (2015).

### 4.2 Reconstructor

The reconstructor back-translates hidden states of the decoder into the source sequence. At each time step $i$, the hidden state $\boldsymbol{h}_i'^{(l)}$ of each layer of the reconstructor is represented as:

$$\boldsymbol{h}_i'^{(l)} = \mathrm{LSTM}(\boldsymbol{h}_i'^{(l-1)}, \boldsymbol{h}_{i-1}'^{(l)}) \quad (25)$$

[3] https://github.com/yukio326/Reconstructor-NMT

| Model | BLEU | RIBES | AMFM | HUMAN |
|---|---|---|---|---|
| Baseline | 24.94 | 0.757955 | 0.596590 | - |
| GAN NMT | 25.17 | 0.757413 | 0.595850 | - |
| Reconstructor NMT | 24.98 | 0.759238 | 0.599110 | - |
| Ensemble of six baselines | **25.85** | **0.761450** | **0.600730** | -20.000 |
| Ensemble of two models each | 25.45 | 0.759790 | 0.598770 | - |

Table 2: Japanese–English translation results.

| Model | BLEU | RIBES | AMFM | HUMAN |
|---|---|---|---|---|
| Baseline | 35.17 | 0.827386 | 0.749190 | - |
| GAN NMT | 35.09 | 0.827650 | 0.750350 | - |
| Reconstructor NMT | 34.89 | 0.826013 | 0.752100 | - |
| Ensemble of six baselines | **36.14** | **0.831219** | **0.753040** | -12.000 |
| Ensemble of two models each | 35.44 | 0.829178 | 0.752420 | - |

Table 3: English–Japanese translation results.

Note that $\boldsymbol{h}_i^{\prime(0)}$ is regarded as the concatenation of the source word embedding vector $\boldsymbol{e}_{i-1}^{s}$ and the attentional hidden state $\tilde{\boldsymbol{h}}_{i-1}^{\prime}$ at the previous time step: $[\boldsymbol{e}_{i-1}^{s} : \tilde{\boldsymbol{h}}_{i-1}^{\prime}]$. In this system, the first hidden state ${\boldsymbol{h}_1}^{\prime(l)}$ of each layer is initialized by the hidden state $\boldsymbol{h}_{|\boldsymbol{Y}|}^{(l)}$ of the decoder.

The attentional hidden state $\tilde{\boldsymbol{h}}_i^{\prime}$ is represented as:

$$\tilde{\boldsymbol{h}}_i^{\prime} = \tanh(\boldsymbol{W}_{a^{\prime}}[\boldsymbol{h}_i^{\prime(L)} : \boldsymbol{c}_i] + \boldsymbol{b}_{a^{\prime}}) \quad (26)$$

where $\boldsymbol{W}_{a^{\prime}} \in \mathbb{R}^{r \times 2r}$ is a weight matrix and $\boldsymbol{b}_{a^{\prime}} \in \mathbb{R}^{r}$ is a bias vector.

The inverse context vector $\boldsymbol{c}_i^{\prime}$ is a weighted sum of each hidden state $\tilde{\boldsymbol{h}}_j$ of the decoder on forward translation. It is represented as:

$$\boldsymbol{c}_i^{\prime} = \sum_{j=1}^{|\boldsymbol{Y}|} \alpha_{ji}^{\prime} \tilde{\boldsymbol{h}}_j. \quad (27)$$

Its weight $\alpha_{ji}$ is a normalized probability distribution, which is computed using the dot product of hidden states as follows:

$$\alpha_{ji}^{\prime} = \frac{\exp(\tilde{\boldsymbol{h}}_j^{\mathrm{T}} \boldsymbol{h}_i^{\prime(l)})}{\sum_{k=1}^{|\boldsymbol{Y}|} \exp(\tilde{\boldsymbol{h}}_k^{\mathrm{T}} \boldsymbol{h}_i^{\prime(l)})}. \quad (28)$$

The conditional probability of the output word $\hat{\boldsymbol{x}}_i$ is computed as:

$$\boldsymbol{p}(\hat{\boldsymbol{x}}_i | \boldsymbol{X}_{<i}, \tilde{\boldsymbol{h}}) = \mathrm{softmax}(\boldsymbol{W}_{p^{\prime}} \tilde{\boldsymbol{h}}_i + \boldsymbol{b}_{p^{\prime}}) \quad (29)$$

where $\boldsymbol{W}_{p^{\prime}} \in \mathbb{R}^{v_s \times r}$ is a weight matrix and $\boldsymbol{b}_{p^{\prime}} \in \mathbb{R}^{v_s}$ is a bias vector.

### 4.3 Training

The objective function is defined as:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\lambda}) = \frac{1}{D} \sum_{d=1}^{D} \left\{ \sum_{j=1}^{|\boldsymbol{Y}|} \log \boldsymbol{p}(\boldsymbol{y}_j^{(d)} | \boldsymbol{Y}_{<j}^{(d)}, \boldsymbol{X}^{(d)}, \boldsymbol{\theta}) + \sum_{i=1}^{|\boldsymbol{X}|} \log \boldsymbol{p}(\boldsymbol{x}_i^{(d)} | \boldsymbol{X}_{<i}^{(d)}, \tilde{\boldsymbol{h}}^{(d)}, \boldsymbol{\lambda}) \right\} \quad (30)$$

where $\boldsymbol{\lambda}$ is the model parameters in the reconstructor. We applied pre-training to the encoder–decoder using the baseline.

## 5 Experiments

We experimented with our NMT system on Japanese–English and English–Japanese scientific paper translation subtasks at the WAT 2018.

### 5.1 Datasets

We used the Japanese–English parallel corpus in the Asian Scientific Paper Excerpt Corpus (ASPEC) (Nakazawa et al., 2016). Japanese sentences were segmented by the morphological analyzer: MeCab[4] (version 0.996, IPADIC) and English sentences were tokenized by tokenizer.perl of Moses[5]. Regarding the training data, we used only the first million

[4]https://github.com/taku910/mecab

[5]http://www.statmt.org/moses/

| | |
|---|---|
| Source | Blood collection is indispensable for glucose level measurement for the diabetes mellitus diagnosis at present. |
| Baseline | 糖尿病診断のためには血糖値測定には採血が不可欠である。 |
| GAN NMT | 現在糖尿病診断のための血糖値測定には採血が必須である。 |
| Reconstructor NMT | 糖尿病診断のための血糖値測定には採血が必須である。 |
| Ensemble of six baselines | 糖尿病診断のための血糖値測定には採血が必須である。 |
| Ensemble of two models each | 糖尿病診断のための血糖値測定には採血が必須である。 |
| Reference | 糖尿病診断のための血糖値測定は，現在，採血が不可欠である。 |

Table 4: Example of outputs of English–Japanese translation.

| | adequacy | fluency |
|---|---|---|
| GAN NMT $>$ Baseline | 16 | 23 |
| GAN NMT $=$ Baseline | 72 | 72 |
| GAN NMT $<$ Baseline | 12 | 5 |
| total | 100 | 100 |

Table 5: Pairwise evaluation between baseline and GAN NMT.

sentences sorted by sentence-alignment confidence; sentences with more than 60 words were excluded. Table 1 shows the number of sentences in the parallel corpus.

### 5.2 Network Settings

We conducted the experiment using the following configuration:

- Number of layers: 3
- Number of hidden units: 512
- Word embedding dimensionality: 512
- Source vocabulary size: 100,000
- Target vocabulary size: 30,000
- Minibatch size: 128
- Optimizer: Adam, SGD
- Initial learning rate: 0.01
- Dropout rate: 0.2
- Beam size: 20

Regarding the optimizer, after we train our model using Adam for 20 epochs, we switch to SGD.

### 5.3 Results

Tables 2 and 3 show the translation accuracy in BLEU (Papineni et al., 2002), RIBES (Isozaki et al., 2010), AMFM (Banchs and Li, 2011), and HUMAN evaluation scores, which are the result of pairwise crowdsourcing evaluation by five different workers at the WAT 2018. In the "Model" column, "Ensemble of two models each" indicates the ensemble of two baselines, two GAN NMTs, and two reconstructor NMTs (ensemble of six models in total).

Regarding Japanese–English translation, the results show that GAN NMT and reconstructor NMT slightly improved BLEU score compared with the baseline. However, in English–Japanese translation, BLEU score of the baseline is higher than the GAN and reconstructor NMTs. In terms of AMFM score, both methods have higher scores than the baseline.

Matsumura et al. (2017) reported that the reconstructor NMT significantly improves BLEU score in English–Japanese translation. This differs from the results in this study. We consider that only by applying the optimization method in this study, the baseline becomes considerably stronger; therefore, the difference of translation accuracy between baseline and reconstructor NMT becomes less.

In both translation subtasks, the ensemble of six baselines achieved the best score in all metrics. The ensemble of two models each is inferior compared with the ensemble of six baselines. The reason for this could be that the ensemble of six baselines considers perfectly independent six models in terms of parameter initialization; however, the ensemble of two models each considers dependent models, i.e., GAN and reconstructor NMTs are pretrained using the baseline. Furthermore, the training of GAN is unstable; therefore, the model that is not trained well may affect the ensemble model adversely.

Table 4 shows an example of outputs of English–Japanese translations. In the baseline, "for the diabetes mellitus diagnosis at present" is translated to "糖尿病診断のためには", but it should be translated to "糖尿病診断のための" when this phrase modify the noun phrase. Other models except GAN NMT slightly under-translate;

“現在 (at present)” is disappeared. However, GAN NMT perfectly translates.

We examine the effect of GAN NMT by the pairwise evaluation between the baseline and GAN NMT. We evaluated 100 sentences extracted randomly in terms of adequacy and fluency. Table 5 shows the numbers of sentence in each case. Regarding adequacy, GAN NMT performed as same as the baseline, but regarding fluency, GAN NMT outperformed the baseline.

## 6 Conclusion

In this paper, we described our NMT system, which is based on the attention-based NMT. Furthermore, we implemented GAN and reconstructor models in our NMT. We evaluated our NMT system on Japanese–English and English–Japanese translation subtasks at the WAT 2018. The experimental results demonstrates that the ensemble of baseline systems achieved 25.85 and 36.14 points in Japanese–English and English–Japanese translations, respectively, in terms of BLEU scores. Furthermore, we found that GAN NMT can translate fluently.

## References


Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR2015)*.

Rafael E Banchs and Haizhou Li. 2011. AM-FM: A Semantic Framework for Translation Quality Assessment. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 153–158, Portland, Oregon, USA. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Automatic Evaluation of Translation Quality for Distant Language Pairs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 944–952, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Yukio Matsumura, Takayuki Sato, and Mamoru Komachi. 2017. English-Japanese Neural Machine Translation with Encoder-Decoder-Reconstructor. *CoRR*, abs/1706.08198.

Toshiaki Nakazawa, Manabu Yaguchi, Kiyotaka Uchimoto, Masao Utiyama, Eiichiro Sumita, Sadao Kurohashi, and Hitoshi Isahara. 2016. ASPEC: Asian Scientific Paper Excerpt Corpus. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 2204–2208, Paris, France, May. European Language Resources Association (ELRA).

Toshiaki Nakazawa, Shohei Higashiyama, Chenchen Ding, Raj Dabre, Anoop Kunchukuttan, Win Pa Pa, Isao Goto, Hideya Mino, Katsuhito Sudoh, and Sadao Kurohashi. 2018. Overview of the 5th workshop on asian translation. In *Proceedings of the 5th Workshop on Asian Translation (WAT2018)*, Hong Kong, China, December.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In Z Ghahramani, M Welling, C Cortes, N D Lawrence, and K Q Weinberger, editors, *Advances in Neural Information Processing Systems 27 (NIPS2014)*, pages 3104–3112. Curran Associates, Inc.

Zhaopeng Tu, Yang Liu, Lifeng Shang, Xiaohua Liu, and Hang Li. 2017. Neural Machine Translation with Reconstruction. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, pages 3097–3103, San Francisco, California, USA.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All

you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Zhen Yang, Wei Chen, Feng Wang, and Bo Xu. 2018. Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets . In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1346–1355, New Orleans, Louisiana, USA. Association for Computational Linguistics.