PREPRINT

PERTANIKA JOURNALS

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

Sliding Window and Parallel LSTM with Attention and CNN for Sentence Alignment on Low-Resource Languages

Tien-Ping Tan[1]*, Chai Kim Lim[1] and Wan Rose Eliza Abdul Rahman[2]

[1]School of Computer Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia
[2]School of Humanities, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia

ABSTRACT

A parallel text corpus is an important resource for building a machine translation (MT) system. Existing resources such as translated documents, bilingual dictionaries, and translated subtitles are excellent resources for constructing parallel text corpus. A sentence alignment algorithm automatically aligns source sentences and target sentences because manual sentence alignment is resource-intensive. Over the years, sentence alignment approaches have improved from sentence length heuristics to statistical lexical models to deep neural networks. Solving the alignment problem as a classification problem is interesting as classification is the core of machine learning. This paper proposes a parallel long-short-term memory with attention and convolutional neural network (parallel LSTM+Attention+CNN) for classifying two sentences as parallel or non-parallel sentences. A sliding window approach is also proposed with the classifier to align sentences in the source and target languages. The proposed approach was compared with three classifiers, namely the feedforward neural network, CNN, and bi-directional LSTM. It is also compared with the BleuAlign sentence alignment system. The classification accuracy of these models was evaluated using Malay-English parallel text corpus and UN French-English parallel text corpus. The Malay-English sentence alignment performance was then evaluated using research documents and the very challenging Classical Malay-English document. The proposed classifier obtained

ARTICLE INFO

Article history:
Received: 7 July 2021
Accepted: 15 September 2021
Published: 24 November 2021

DOI: https://doi.org/10.47836/pjst.30.1.06

E-mail addresses:
tienping@usm.my (Tien-Ping Tan)
chaikimlim@gmail.com (Chai Kim Lim)
wardah@usm.my (Wan Rose Eliza Abdul Rahman)
*Corresponding author

ISSN: 0128-7680
e-ISSN: 2231-8526

more than 80% accuracy in categorizing parallel/non-parallel sentences with a model built using only five thousand training parallel sentences. It has a higher sentence alignment accuracy than other baseline systems.

Keywords: Attention, CNN, LSTM, parallel text, sentence alignment

INTRODUCTION

A parallel text corpus is an important resource for building a machine translation (MT) system containing words, phrases, or sentences of two or more languages aligned semantically. An example is the UN parallel text corpus created from official records and parliamentary documents of the United Nations in 6 languages (Ziemski et al., 2016). Table 1 shows a snippet of the English-French parallel text entries. The second column consists of an English word, phrase, or sentence and their corresponding translation in column 3. For instance, entry #1 is the English word "GENERAL," translated as "*GÉNÉRALE*" in French. Entry #3 is an English phrase "2. Paragraphs 4, 5 and 6 of the resolution read as follows", which was translated in French as "*2. Les paragraphes 4, 5 et 6 de cette résolution se lisent comme suit*."

Table 1
UN Parallel Text Corpus (English-French)

#	English	French
1.	GENERAL	GÉNÉRALE
2.	2 February 1999	2 février 1999
3.	2. Paragraphs 4, 5 and 6 of the resolution read as follows:	2. Les paragraphes 4, 5 et 6 de cette résolution se lisent comme suit :
4.	In the example of approval marks and in the captions below, replace approval number "001234" by "011234".	Dans les exemples de marques d'homologation et dans les légendes situées en dessous, remplacer le numéro d'homologation "001234" par "011234".

A parallel text corpus can be constructed manually or automatically. Normally, a parallel text corpus is only manually created for a language pair if existing resources such as translated documents, bilingual dictionaries are not available. For example, Almeman et al. (2013) described an effort to collect a parallel Arabic dialects corpora consisting of parallel sentences and speech utterances in Modern Standard Arabic, Gulf, Egypt, and Levantine dialects. The Modern Standard Arabic text, consisting of more than a thousand sentences, was prepared before being translated to the other three dialects. Another example

is the Malay dialect parallel corpora (Khaw et al., 2021) that contain the Kelantan Malay-Standard Malay and Sarawak Malay-Standard Malay parallel text and speech utterances. The dialect speech was first recorded. The speech was then transcribed before it was translated to Standard Malay.

If a parallel or comparable text is available, parallel sentences can be extracted to construct a parallel text corpus. A text is segmented into a smaller unit first, generally in sentences. For translated documents, such as novels and technical reports, after the text is segmented, the sentences in the source language text and the sentences in the target language text can be aligned to produce a parallel text corpus using a sentence alignment system.

Sentence Alignment

The sentence alignment system automatically aligns sentences in the source language text and the target language text. The goal of sentence alignment is to match the source and target sentences with similar meanings. The sentence alignment algorithms may use the similarity comparison in terms of the sentence length and co-occurrence of lexical items in the source and target sentences to determine the sentences aligned together.

The early sentence alignment algorithms used sentence length as the heuristics for aligning sentences. Brown et al. (1991) proposed to use the number of words in the sentences for alignment. The basic idea is to align a long source sentence with a long target sentence and vice versa. Another heuristic used in the alignment is that the alignment must be in a monotonic sequence. For example, if source sentence i is aligned to target sentence j, source sentence i+1 cannot be aligned to sentence j-1. The approach aligned source sentences to target sentences with many-to-one or one-to-many relationships, but at most, only two sentences can align with one sentence. The algorithm also used anchor points to reduce the complexity of the alignment algorithm. Gale and Church (1993) also proposed to use sentence length for aligning sentences. However, the approach is based on the character statistics in the sentence. A probabilistic score was calculated for each sentence pair based on the scaled difference of length of the sentence pair and the variance of the difference. Finally, the maximum likelihood alignment of the sentences was calculated to find the best alignment.

The limitation of the previous approaches is that they ignore the lexical items in sentences. Chen (1993) proposed a statistical word-based translation model that captures the word alignment probability of a source word w_s and a target word w_t, $p(w_s, w_t)$. The expectation-maximization (EM) algorithm was used to estimate the model. The proposed approach calculated the sentence alignment probability for 1:0, 0:1, 1:1, 2:1, and 1:2 sentences. Sentence alignments with the highest probability were selected, and a minimum threshold was also applied to remove possible sentences without alignment. Another example is Champollion, which is a Chinese-English sentence alignment system. The

approach requires a bilingual word lexicon to be prepared before alignment (Ma, 2006). The bilingual lexicon was created from Chinese-English bilingual dictionaries. Champollion uses the bag-of-words model, and it calculates tf-idf for the words in the source and target segments to determine the similarity of the two segments. A segment consists of one or more sentences. A similarity measure that is based on tf-idf was proposed to calculate the weight of source-target word pairs. The word pairs that appear very frequent but not commonly found in all documents have a higher weight than less frequent and common word pairs. Champollion allows one-to-many and many-to-one alignments, with a maximum of four sentences aligned to one sentence. Dynamic programming was used in the lexical approach to search for the alignments with the highest total similarity.

Sennrich and Volk (2010) proposed an interesting idea to align source sentences and target sentences with the help of an MT system. The MT system was used to generate the translation for the source sentences; the hypothetical target sentences produced were compared with the target sentences using the BLEU metric. BLEU is an evaluation metric used in machine translation to compare the hypothesis translation to the reference translation. The BleuAlign algorithm created a matrix that contains the similarity score in BLEU between the hypothetical target sentence (from the translation of the source sentence) and the target sentences. Sentence pairs with the highest BLEU scores and which appeared in monotonic sequence were selected. Besides, heuristics were used to improve sentence alignment, such as aligning the unaligned sentences using the length-based alignment algorithm. A similar idea was proposed by Wolk and Marasek (2014) that also used MT to create hypothetical target sentences. In addition, they used WordNet to obtain synonyms for words in the target sentences to generate similar target sentences. Finally, a custom similarity metric was proposed that scored the source sentence and target sentences.

Grégoire and Langlais (2017) proposed to use bidirectional recurrent neural networks to extract parallel sentences from Wikipedia. The deep neural network processes a parallel document and outputs sentence alignments. The proposed approach needs a seed parallel text corpus that serves as positive examples for training the classifier. The negative examples were generated by randomly pairing non-parallel sentences. The source and target languages were encoded using bidirectional recurrent neural networks, and the matching information is estimated using their element-wise product and absolute element-wise difference. The cell can be an LSTM or a GRU. The probability that two sentences were a translation of each other was estimated by feeding the matching vectors into fully connected layers. A sentence pair was classified as parallel if the probability score was greater than or equal to a decision threshold.

On the other hand, Luo et al. (2021) presented an unsupervised sentence alignment method using deep neural networks. First, the bilingual pseudo documents were created from parallel documents (Vulić & Moens, 2015). Next, bilingual word embeddings were

extracted using word2vec approaches from pseudo documents. Then, word similarity was calculated using the cosine similarity. Finally, the similarity between two sentences was defined based on word similarity and word position. The sentence alignment problem was then converted into an extended earth mover's distance (EMD) problem. The approach is interesting as it does not require a seed parallel corpus. However, the approach will require a sufficiently large parallel document to learn robust bilingual word embeddings.

Sentence alignment techniques have improved from just sentence length to word translation models and bilingual lexicon. In addition, neural networks are now being used to model word semantics, encoding sentences, and latent word alignments in sentences.

Text Classifiers

The approach using a classifier to align sentences by Grégoire and Langlais (2017) is interesting as classification is the core of machine learning, and the classification algorithms have been improving over the years. Classification is a process that predicts the category/label of a given data. Classification is supervised learning, where the classification algorithm identifies the significant features in the training data that are important in predicting the category of the data. In-text classification, the data being classified is text. The categories that a classifier predicts are task-dependent and depend on the training data. For instance, given a text with sentiment annotation, a classifier will learn to predict the sentiment in a text (Lim et al., 2020). In the case of sentence alignment, the alignment of two sentences of different languages can be solved as a text classification problem. The classifier learns the significant features in two sentences of different languages to predict whether the given sentences are parallel.

Before classification is carried out on a text, the text is first pre-processed, segmented, tokenized, and normalized. The pre-processing step removes the unwanted noise in the text. Next, the sentence segmentation splits a text into sentences. The tokenization step then segments every sentence into tokens that consist of words, punctuations, and numbers. Then, the normalization process standardizes the tokens in the text, such as lowercasing the characters and standardizing the acronyms. Additional steps may be performed depending on the classification problems, such as removing the stop words and lemmatizing/stemming the words.

Next, features are extracted from the normalized text. The types of features used in text classification can be divided into two: bag-of-words and word embedding. The bag-of-words features are used in conventional text classification algorithms such as decision trees, naïve Bayes, support vector machines, and multilayer perceptron. In the bag-of-words approach, a vector is used to store the statistics of every token that appears in the text. The statistics of the words used can be Boolean, word frequency, word probability, and tf-idf. The advantage of the approach is that it is simple to implement, but the disadvantage is

that the contextual information of a word is lost when words are converted to bag-of-words features.

On the other hand, word embedding approaches are techniques where words are represented as real-valued vectors in a vector space. Word embeddings are based on the idea of distributed semantics. The semantic similarities between words are based on their distributional properties in large text samples in distributional semantics. In other words, words that appear in the text with similar contexts have similar semantics. For example, the word "cough" and "sofa" can appear in similar contexts in a sentence. Word embedding is one of the most important advancements in semantic modeling from deep neural networks. Examples of approaches in word embeddings are 1) word embedding layer, 2) word2vec (Mikolov et al., 2013), and 3) count-based word embeddings (Stratos et al., 2015). An embedding layer is the input layer of a neural network that is jointly trained with a natural language processing task, such as sentiment analysis. The word2vec approach, on the other hand, is a standalone neural network approach that learns to convert words to vectors from a large text corpus. Examples of word2vec approaches are Skip-gram and Continuous Bag-of-Words (CBOW). The count-based word embedding is similar to the word2vec approach, where it is a standalone approach and learns from a text corpus. One major difference is that the count-based approach is based on counting the co-occurrence of words. On the other hand, the word2vec approaches do not involve counting words to derive the word to vector mappings.

After a text is converted to the respective features, the classification algorithm is applied to the features. This section discusses a conventional feedforward neural network and two state-of-the-art text classification algorithms: convolutional neural networks (CNN) and bi-directional long short-term memory (LSTM).

Feedforward Neural Network Classifier. Feedforward neural network is the earliest neural network used in classification. Figure 1 shows an example feedforward neural network that processes three inputs and produces two outputs. The output neurons convert the scores s_i to the estimated probabilities y_j of the corresponding classes. Equation 1 corresponds to the input layers, while Equation 2 corresponds to the output layer. The gradient descent algorithm will estimate the weight matrices during training by minimizing the loss function.

$$h = \phi(xW^T + b) \quad [1]$$

$$y = hU^T + b \quad [2]$$

where x is the inputs, y is the outputs, ϕ is the activation function such as ReLU, W and U are the weight matrices, and b is the bias.

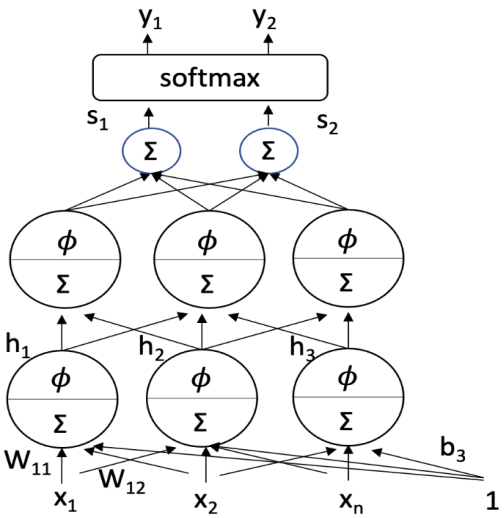

Figure 1. Feedforward neural network with two hidden layers and a Softmax output layer

Bi-directional Long Short-Term Memory Classifier. A recurrent neural network (RNN) is like the feedforward neural network, except it has a feedback loop. Long short-term memory (LSTM) is a special type of recurrent neuron. Bi-directional LSTM consists of a layer of LSTM that processes input from left to right and another layer of LSTM that processes the same input in the opposite direction. Many studies showed that bi-directional LSTM gives good results in time series classification problems such as sentiment analysis and text classification (Zhou et al., 2016). Figure 2a shows an LSTM memory cell. x_t is an input vector; σ and tanh are neural network layers with sigmoid and hyperbolic tan activation function respectively, refer to Equation 1, c_t is the current cell state; h_t is the hidden state; x and + are the pointwise multiplication and addition operation respectively. Figure 2b shows that input vectors x_t is input to a layer of forward LSTM, $LSTM_{fw}$ that processes the input vectors from left to right, and a layer of backward LSTM, $LSTM_{bw}$ that processes the input in the opposite direction. Refer to Equations 3, 4, and 5.

$$h_{fw,t} = LSTM_{fw}\left(h_{fw,t-1}, x_t\right) \quad [3]$$

$$h_{bw,t} = LSTM_{bw}\left(h_{bw,t-1}, x_t\right) \quad [4]$$

$$h_t = \left[h_{fw,t}; h_{bw,t}\right] \quad [5]$$

The output states, h_t from both LSTM are concatenated to become a single vector, and the vector goes through a Softmax output layer before classification. One or more dense layers are often added before the output layer.

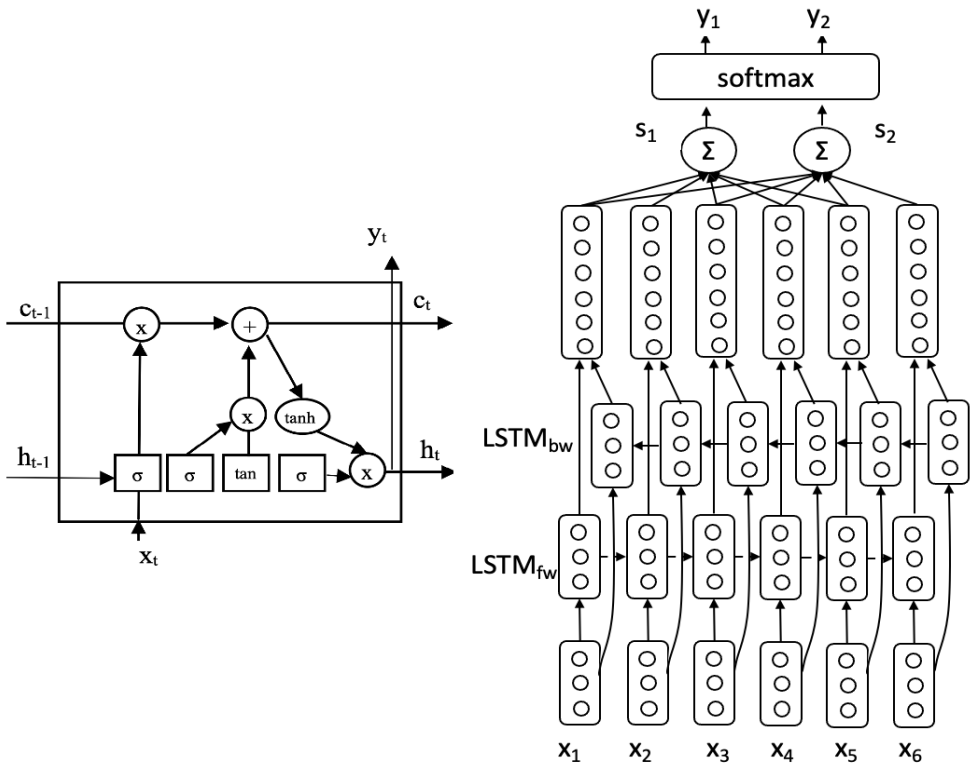

Figure 2. a) Left, an LSTM memory cell. b) Right, a bi-directional LSTM with a SoftMax output layer

Convolutional Neural Network Classifier. Convolutional neural network (CNN) emerged from the study of image recognition. Since its introduction, it has achieved state-of-the-art performance in image recognition tasks and later text classification (Kim, 2014). A CNN consists of a convolutional layer and pooling layer. A convolutional layer consists of filters or kernels. The purpose of a filter is to highlight the areas in an image that are most similar to it. The weights for a filter will be learned during training, and the networks learn to combine the filters to recognize complex patterns (Equation 6).

$$h_{i,j,k} = \phi\left(\sum_{u=1}^{h}\sum_{v=1}^{w}\sum_{k=1}^{n} x_{i\prime,j\prime}w_{u,v,k} + b_k\right) \quad [6]$$

where $h_{i,j,k}$ is the output of the feature map k, at row i, and column j; w is the weights of the filter k; b_k is the bias of feature map k.

The pooling layer is a downsampling operation that is applied normally after the convolutional layer. There are two types of pooling functions, max-pooling and average-pooling. The max-pooling operation selects the maximum value of the current view, while the average-pooling averages the values of the current view. Max-pooling preserves the detected features, and it is more commonly used.

Figure 3 shows a CNN used for recognizing a written character. It consists of a convolutional layer and a max-pooling layer. The output of a max-pooling has to be flattened to become a one-dimensional vector before it can be input to a layer of dense network for classification.

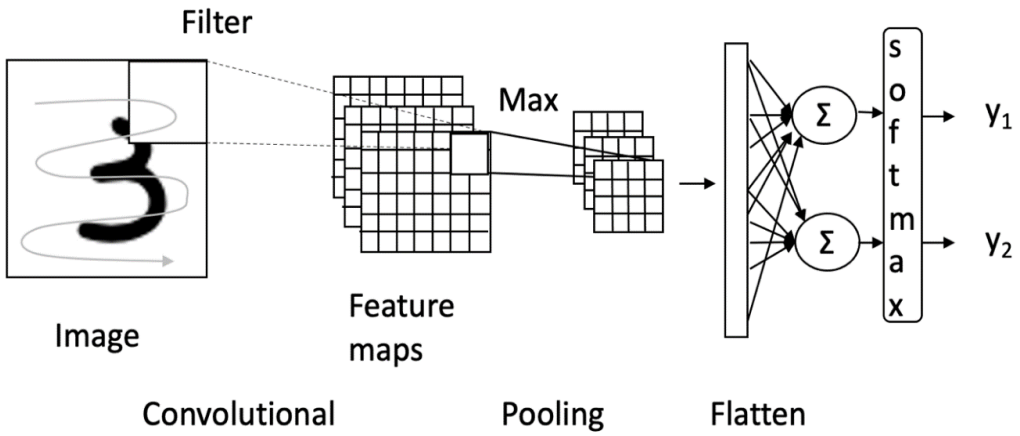

Figure 3. CNN classifier for image recognition. A convolutional layer is followed by a max-pooling layer and a Softmax output layer

MATERIALS AND METHODS

We solve the problem of aligning two sentences of different languages as a classification problem. The Boolean classifier scores two sentences as parallel/non-parallel sentences given text in the source and the target languages. A sliding window subsequently aligns the source sentences and the target sentences using the scores generated by the classifier.

Parallel LSTM with Attention and CNN Classifier

We propose a new model, parallel LSTM with attention and CNN, classifying a source sentence and a target sentence as parallel/non-parallel sentences. Thus, the classifier will evaluate a given pair of semantically similar sentences as a parallel sentence and a pair of semantically non-similar sentences as non-parallel. For example, the pairs of words/phrases/sentences in Table 1 are parallel because they are semantically similar. Figure 4

shows our proposed architecture. The encoder-decoder inspires the proposed architecture with attention architecture used in sequence-to-sequence modeling (Luong et al., 2015).

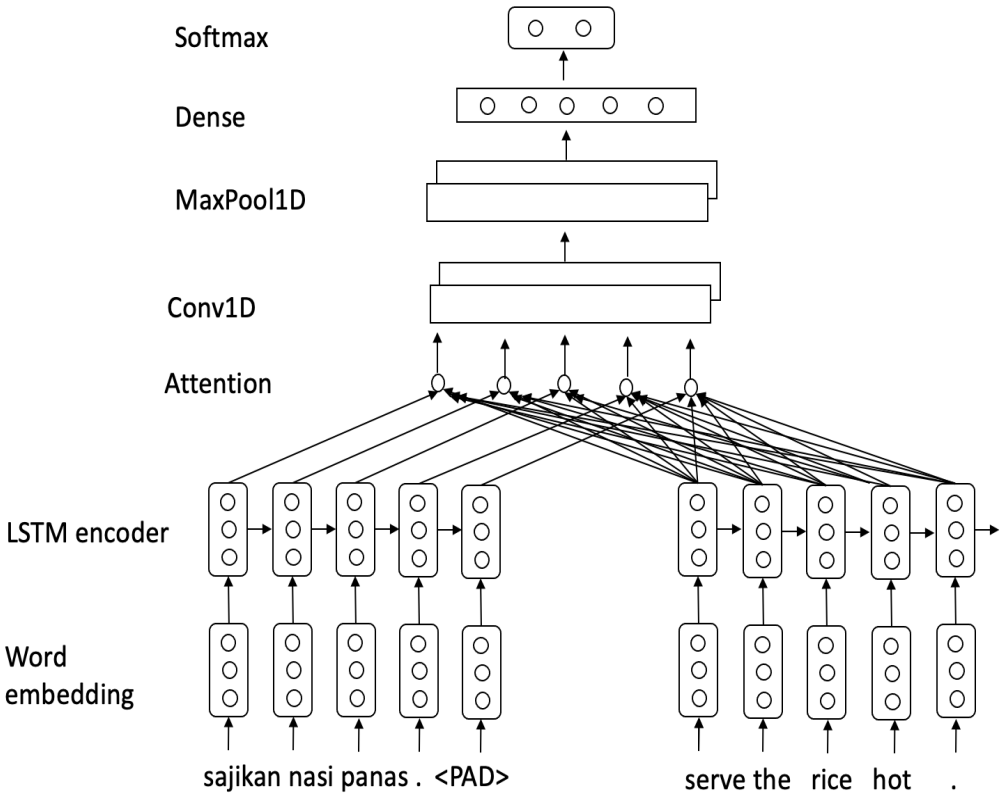

Figure 4. Parallel LSTM with attention and CNN (Parallel LSTM+Attention+CNN)

The sentences will be tokenized and normalized given a source sentence, $x_{s,}$ and a target sentence, x_t. The tokens in the sentences will be converted to embedding vectors, $v_{s,}$ and v_t. In the proposed approach, we use the word embedding model. First, the word2vec approach is preferred since it reduces the time for training the classifier model later. Second, the number of parameters to be tuned during training is reduced. Third, more text can train the word embedding vectors, producing a more robust model. The vocabulary for the word embedding model must be specified. The words that are not in the vocabulary will be mapped to a special tag, <UNK>. The length of a sentence is set to N tokens. If the length of a sentence is less than N, then the sentence will be padded with the tag <PAD>.

After the word embedding models for the source and target languages have been trained, the source sentences and target sentences are converted to the embedding vectors

using the respective model. The source vector v_s is input to a source LSTM, while the target vector v_t is input to a target LSTM. For each input vector, the LSTM will output a hidden state. For example, the source LSTM will output hidden states h_i, while the target LSTM will output hidden states s_i. The attention scores are calculated using Equation 7 below. In general, the attention mechanism calculates the alignment distribution between the source output hidden states and the target output hidden states. For example, if the length of the source sentence and the target sentence is N, then an attention matrix of size NxN will be produced.

$$e(t,i) = [s(t)^T, h(i=1)\ s(t)^T h(2)\ \ldots, s(t)^T\ h(N)] \quad [7]$$

The attention matrix is input to a layer of a convolutional layer. The purpose of the convolutional layer is to extract high-level features from the attention scores that are important in identifying the semantic similarity between the source sentence and the target sentence. Max-pooling, on the other hand, filters the noise in the data by choosing the prominent features. The matrix output will be flattened before input to a dense layer of neurons. The output from the dense layer will then be sent to the Softmax layer containing two neurons classified as non-parallel or parallel. The Softmax layer gives a probability/ score between zero and one for each category. Since there are only two categories for this model, the category that has a probability of more than 0.5 is selected.

For training the classifier to learn to identify non-parallel/parallel sentences, training examples from a seed parallel text corpus is required. Since a parallel text corpus only contains the valid pairs of the parallel sentence, the non-parallel sentence pairs (or semantically non-similar sentence pairs) have to be added to the training data. We generate the non-parallel sentence pairs by pairing for every source sentence a randomly selected target sentence where the sentence length is plus/minus three tokens. If no target sentence meets the requirement can be found, then a random target sentence is paired with the source sentence. The purpose of selecting a sentence with a length close to the valid target sentence is to avoid the classifier from using the sentence length as a criterion to identify non-parallel/parallel sentences and to force it to learn from the similar lexical items in the sentences. Therefore, the parallel pairs are annotated as “1” and the non-parallel pairs as “0”. Table 2 shows an example of generated records from the data in Table 1.

Table 2

An example of generated records from the data in Table 1

English	French	Ann.
GENERAL	GÉNÉRALE	1
GENERAL	2 février 1999	0
2 February 1999	2 février 1999	1
2 February 1999	GÉNÉRALE	0
2. Paragraphs 4, 5 and 6 of the resolution read as follows:	2. Les paragraphes 4, 5 et 6 de cette résolution se lisent comme suit :	1
2. Paragraphs 4, 5 and 6 of the resolution read as follows:	2 février 1999	0
In the example of approval marks and in the captions below, replace approval number “001234” by “011234”.	Dans les exemples de marques d’homologation et dans les légendes situées en dessous, remplacer le numéro d’homologation “001234" par “011234".	1
In the example of approval marks and in the captions below, replace approval number “001234” by “011234”.	2. Les paragraphes 4, 5 et 6 de cette résolution se lisent comme suit :	0

Source Sentence and Target Sentence Sliding Window

The information on the type of document for sentence alignment can be taken advantage. When the sentences to be aligned are translated texts, for example, the translation of a novel or book, most of the sentences, if not all, in the source text and the target text should be aligned in the same order. Nevertheless, it is possible for a translator to translate a part of a sentence or to translate more than one sentence into a single sentence. Therefore, the heuristics that the source sentences and target sentences appear in the monotonic sequence can be used in aligning the sentences assuming the texts we are aligning are translated materials. Besides, by knowing the type of documents used in alignment, the time complexity of the alignment can be improved by searching the areas for possible targets instead of searching for the target everywhere.

We propose to use a sliding window that will match a source sentence at line i, $x_{s,i}$ to some target sentences at line j, $x_{s,j}$. The classifier is then used to classify the pair of the source sentence and target sentence as either non-parallel/parallel. Below is the definition of the sliding window (Equations 8, 9, and 10):

$$i' = i + 1 \quad [8]$$

$$j' = \text{round}(i' \times J) \quad [9]$$

$$J = N_t/N_s \quad [10]$$

$x_{s,i}$: source sentence at line i. $0<i \leq N_s$
$\{x_{s,j}$: target sentences at line j, $i-d \leq j \leq i+d \cap 0<j \leq N_t \}$
i–the current source sentence at line i.
i'–the next source sentence at line i'.
j'–the next target sentence at line j'.
J–incrementing step. $J>0 \cap J \in R$.
d–size of the sliding window. $d \in \mathbb{Z}^{0+} \cap d < N_t$.
N_s–the total number of lines in the source text.
N_t–the total number of lines in the target text.

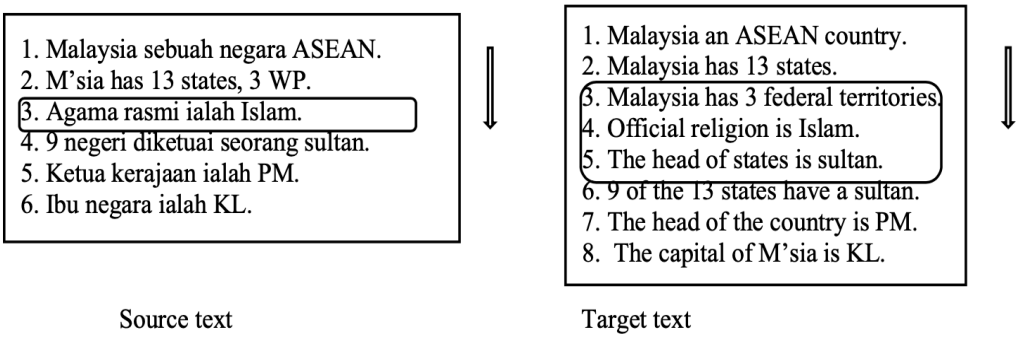

Figure 5. A sliding window matches the source sentence at line three to target sentences at lines three, four, and five, where d=1, J=1.33

Figure 5 shows a source text with six sentences and a target text with eight sentences, and the sliding window is based on the definition given in Equations 8, 9, and 10. Two important parameters that have to be set for the sliding window are d, and J. d is manually set, normally in the range of 1 to 5. Step J can be manually set or set by default calculated as in Equation 10. In Figure 5, the source sentence at line 3 is matched to the target sentences at lines 3, 4, and 5. The classifier is then used to classify every pair of a source-target sentence, $(x_{s,3}, x_{t,3})$, $(x_{s,3}, x_{t,4})$, and $(x_{s,3}, x_{t,5})$.

For aligning the sentences in the source text and the target text, we use the probability returned by the classifier when it compares a given source sentence $x_{s,i}$ and a target sentence $x_{t,j}$, for classification, instead of using the classification category that is returned by the classifier. Given a source text and a target text, the source sentence, and the target sentence alignment, A' is the set of the source sentence and target sentence alignment, and $a'(x_{s,i}, x_{t,j})$ is the tuple that consists of the aligned source sentence and target sentence such as in Equation 11:

$$A' = argmax\left(\sum_{\substack{0 \leq i \leq N_s \\ 0<j<N_t}} P(x_{s,i}, x_{t,j})\right) \quad [11]$$

where $x_{s,i}$ is line/sentence i of source text, and $x_{t,j}$ is line/sentence j of target text. N_s is the total number of lines in the source text, and N_t is the total number of lines in the target text. If $a'(x_{s,i}, x_{t,j})$, $a'(x_{s,p}, x_{t,q})$ and if $p >= i \rightarrow q >= j$. In other words, we want to find the source-target sentence alignment, A', that has the highest overall probability. A source sentence can align to one or more target sentences, and a target sentence can be aligned to one or more source sentences, but the alignments must be in a monotonic sequence.

Dynamic programming is used to find the set of aligned source sentences and target sentences that has the highest overall probability, $a'(x_{s,i}, x_{t,j})$. A table is used to store the score of the alignments, $Sc(x_{s,i}, x_{t,j})$. The example source text and target text from Figure 5 are used to create Figure 6. The bottom first row of Figure 6 indicates the sentence i of the source text, while the first column indicates the sentence j of the target text. A cell in the table stores the alignment probability between a source sentence i and a target sentence t, $P(x_{s,i}, x_{t,j})$. The probability $P(x_{s,i}, x_{t,j})$ is converted to log probability to improve the time complexity and to avoid underflow. Thus, the $P(x_{s,i}, x_{t,j})$ in Equations 12 and 13 are in log probability. Imaginary $P(x_{s,i}, x_{t,j})$ were assigned in Figure 6 to show the calculations. The grey cells correspond to source-target sentence pairs that were not evaluated by the classifier because they are outside of the sliding window. Thus, the log probability $P(x_{s,i}, x_{t,j})$ is set to $-\infty$. The algorithm calculates from the bottom left cell to the top right cell. The score for the cell (i=1,j=1) is initialize to the value of $P(x_{s,1}, x_{t,1})$, while the alignment is set to (0, 0) (Equations 12 & 14). Equation 13 is used to calculate the rest of the cells. For example score at the cell (2,2) is calculated as follow:

$$\begin{aligned} Sc(x_{s,2},x_{t,2}) &= \max(Sc(x_{s,1}, x_{t,2}), Sc(x_{s,2}, x_{t,1}), Sc(x_{s,1}, x_{t,1})) + P(x_{s,2}, x_{t,2}) \\ &= \max(-1.1, -\infty, \underline{-0.1}) + (-0.2) \\ &= -0.1 -0.2 \\ &= -0.3 \end{aligned}$$

The $a(x_{s,i}, x_{t,j})$ keep track of the alignment i and j that produces the highest score, $Sc(x_{s,i}, x_{t,j})$. Refer to Equation 15. After all the scores are calculated from bottom left to top right, A' can be obtained by backtracking from $a(x_{s,Ns}, x_{t,Nt})$ until $a(x_{s,1}, x_{t,1})$ by following along $a(x_{s,i}, x_{t,j})$ from the top right cell.

8						P=-2 Sc=max(-∞,-3.3,-∞) =**-5.3** a=(5,7)	P=-0.5 Sc=max(-5.3,-4.3,-3.3) =**-3.8** a=**(5,7)**
7						P=-0.1 Sc=max(-∞,-5.7,-3.2)-0.1 =**-3.3** a=**(4,6)**	P=-1 Sc=max(-3.3,-∞,-5.7)-1 =**-4.3** a=(5,7)
6					P=-0.5 Sc=max(-∞,-2.7,-4.7) =**-3.2** a=**(4,5)**	P=-3 Sc=max(-3.2,-∞,-2.7)-3 =**-5.7** a=(4,6)	
5				P=-4 Sc=max(-∞,-0.7,-3.7)-4 =**-4.7** a=(3,5)	P=-2 Sc=max(-4.7,-3.2,-0.7)-2 =**-2.7** a=**(3,4)**		
4			P=-3 Sc=max(-∞,-0.7,-∞)-3 =**-3.7** a=-(2,3)	P=0 Sc=max(-3.7,-2.3,-0.7)+0 =**-0.7** a=**(2,3)**	P=-2.5 Sc=max(-0.7,-∞,-2.3)-2.5 =**-3.2** a=(3,4)		
3			P=-0.4 Sc=max(-∞, -0.3,-1.1) -0.4 =**-0.7** a=**(2,2)**	P=-2 Sc=max(-0.7,-∞,-0.3) - 2 =**-2.3** a=(2,2)			
2		P=-1 Sc=max(-∞, -0.1, -∞) -1 = **-1.1** a=(1,1)	P=-0.2 Sc=max(-1.1, -∞, -0.1) -0.2 = **-0.3** a=**(1,1)**				
1		P=-0.1 Sc=-0.1 a=**(0,0)**					
0							
j i	0	1	2	3	4	5	6

Figure 6. A table is used to keep track of the alignment score, Sc, in dynamic programming. This table is created based on the sliding window in Figure 5.

Backtracking from the top right cell (6,8) in Figure 6, we get (5,7), (4,6), (4,5), (3,4), (2,3), (2,2), (1,1), (0,0), which correspond to the sentence alignment in Table 3.

Table 3

An example of generated records for training the classifier from the data in Table 1

i	Source Text	j	Target Text
1	Malaysia sebuah negara ASEAN.	1	Malaysia an ASEAN country.
2	M'sia has 13 states, 3 WP.	2	M'sia has 13 states, 3 WP.
2	M'sia has 13 states, 3 WP.	3	Malaysia has 3 federal territories.
3	Agama rasmi ialah Islam.	4	Official religion is Islam.
4	9 negeri diketuai seorang sultan.	5	The head of states is sultan.
4	9 negeri diketuai seorang sultan.	6	9 of the 13 states have a sultan.
5	Ketua kerajaan ialah PM.	7	The head of the country is PM.
6	Ibu negara ialah KL.	8	The capital of M'sia is KL.

RESULTS AND DISCUSSIONS

We evaluated our proposed classifier for classifying tasks using two datasets, our Malay-English parallel text corpus and the French-English parallel sentences from UN parallel text corpus (Ziemski et al., 2016). The setup for training and testing the classifiers for both datasets is as follows. The training set consists of 50 thousand parallel sentences, and the validation set consists of 3 thousand parallel sentences. Since the parallel text corpus consists of only valid pairs of parallel sentences, for every source sentence in the parallel text, a pair of the non-parallel sentence was randomly generated as described in the previous section for the training set and validation set. As a result, the size of the training and validation set double. For testing the classifier performance, a test set that consists of more than 25 thousand parallel/non-parallel sentences that were prepared just like the training and validation set was used.

For evaluating the classifier in aligning sentences, only Malay-English texts were evaluated. For testing the classifier for aligning sentences, we prepared two tasks. The first task was to evaluate the alignment of sentences in research documents. We selected ten postgraduate research thesis documents http://eprints.usm.my/view/type/thesis.html that contain abstracts written in Malay and English from different domains such as social science, computer science, and engineering. The abstract texts were selected so that the number of sentences in the source and target documents differed. We combined all the selected texts into a single Malay document and an English document to increase the difficulty further. The resulting Malay document contains 161 sentences, while the English document contains 164 sentences. We aligned sentences in a Classical Malay document

and its translation text in English for the second alignment task. The reason for selecting Classical Malay for alignment is because the result obtained serves as a lower bound for the sentence alignment algorithms as Classical Malay is very different from Standard Malay text. We manually extracted chapter one of "*Hikayat Hang Tuah*" (Ahmad, 2017) and the literature's English translation "The Epic of Hang Tuah" (Salleh, 2010) for evaluation. The extracted Classical Malay document contains 227 sentences, while the English document contains 184 sentences. Table 4 summarizes the data used for training, validation, and testing.

Table 4

Training, validation and testing data

Data	Size (number of sentences)
Training: Malay-English and French-English (UN)	100,000
Validation: Malay-English and French-English (UN)	6,000
Classification Test: Malay-English and French-English (UN)	25,000
Sentence Alignment Test 1: 10 research articles	161 (Malay), 164 (English)
Sentence Alignment Test 2: *The Epic of Hang Tuah*	227 (Classical Malay), 184 (English)

Our proposed parallel LSTM classifier with attention and CNN was trained using the word embedding features. The LSTM layer consists of 128 units. The time step of the LSTM was set to the maximum length of the sentence, which is 80 words. The subsequent layer, which is the convolutional layer, consists of 128 filters, and the size of the filter is 3. The dense layer before the output layer consists of 16 neurons with a ReLU activation function. The input for the classifier is GloVe word embedding features (Pennington et al., 2014). The English word embedding was downloaded from the GloVe GitHub page, and the size of the vocabulary is 400 thousand with 100 dimensions. The Malay GloVe word embedding with 100 dimensions and the vocabulary size was 146 thousand words; It was trained using our Malay text corpus that consists of more than 800 MB of clean text. The French GloVe word embedding with 100 dimensions and more than 350 thousand vocabularies was trained using the French text from the UN parallel text corpus. An unknown word embedding, <UNK> was calculated for the source and target languages by averaging all the word embeddings for each corresponding language. The <PAD> word embedding vector was set to zeros.

We compared our proposed classifier with three baseline classifiers. First, a conventional text classifier using the feedforward neural network and bag-of-words model was evaluated. The feedforward neural network consists of two hidden layers. The first layer consists of 128 neurons, while the second layer consists of 32 neurons. The activation function for the neurons is ReLU, and the optimizer was Adam. Our dataset's source and target sentences were tokenized using NLTK tokenizer and normalized by lowercasing the characters. The stop words were removed from the sentences. The words not in the vocabulary list were mapped to the <UNK> in the source/target language. The tf-idf for the words in the source sentences and target sentences were calculated separately for the source language and target language. Finally, the bag-of-words vector from the source sentence and the target sentence were concatenated to become one. The feedforward neural network with the bag-of-words model was trained using the training set and the above validation set until converged.

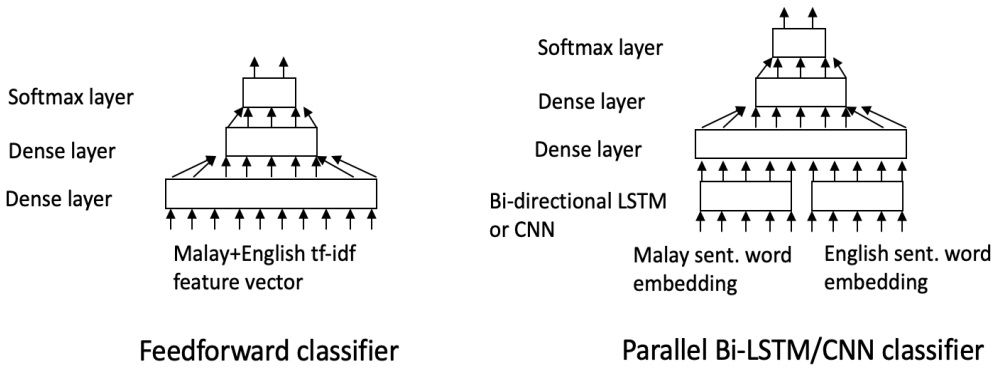

Figure 7. The architecture of the feedforward classifier, parallel bi-directional LSTM, and parallel CNN classifier used.

Two state-of-the-art classifiers using bi-directional LSTM and CNN discussed in the previous section were also compared to the proposed model. Some modifications allowed the classifiers to process two inputs, Malay-English or French-English sentences/word embedding vectors. Refer to Figure 7. The parallel bi-directional LSTM classifier consists of two LSTMs that process Malay and English word embedding feature vectors. Both bi-directional LSTM was set to 128 units. The time step was set to 80. The outputs from the two LSTMs were flattened and concatenated to become a single vector before sending it to two-layer dense neural networks with 64 and 16 neurons, respectively, with ReLU activation function. Finally, the output from the dense layer was sent to the Softmax output layer. For the parallel CNN classifier, two CNNs of the convolutional layer and the max-pooling layer were used to process Malay and English word embedding features. The convolutional layer

consists of 128 filters with the size set to 3. Like the bi-directional LSTM, the CNN layer output was flattened and sent to a two-layer dense neural network before the output went to the Softmax output layer. The number of neurons at the two-layer dense neural network was set to 32 and 8, respectively, with the ReLU activation function. The optimizer used was Adam. The models were trained until they converged.

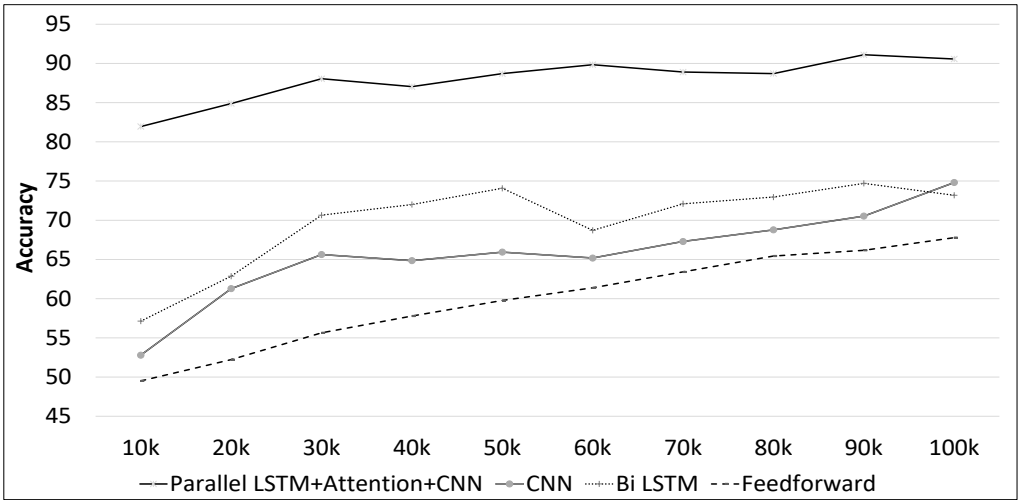

Figure 8. Malay-English parallel/non-parallel sentence classification using different classifiers with different amounts of training data.

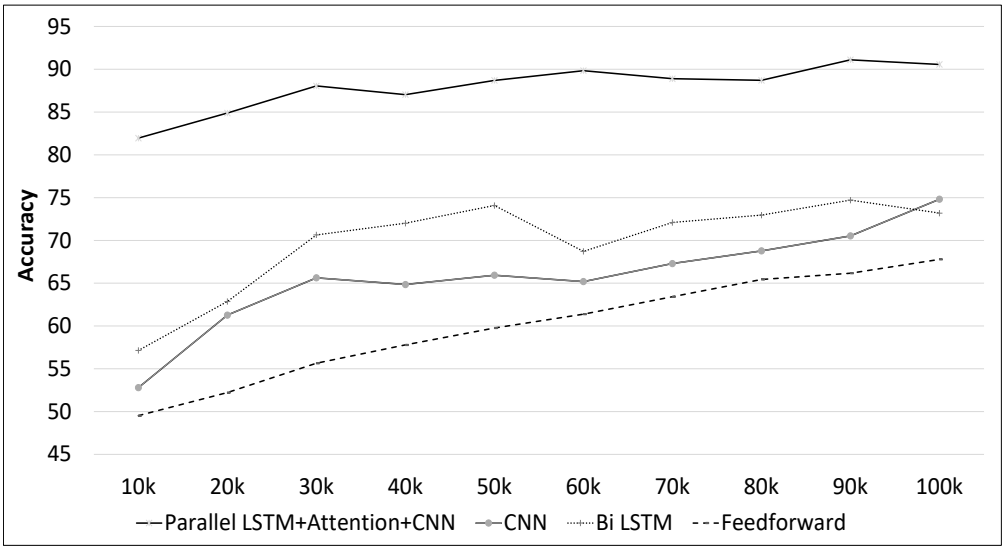

Figure 9. UN French-English parallel/non-parallel sentence classification using different classifiers with different amounts of training data.

Figures 8 and 9 show parallel/non-parallel sentence classification accuracy using different Malay-English models and French-English models. In both the experiments, the training data used to build the models varies from 10 thousand to 100 thousand sentences. The results show that our proposed parallel LSTM+Attention+CNN classifier achieved the highest accuracies under different settings, followed by parallel bi-directional LSTM, parallel CNN, and feedforward neural network in both datasets. The classifier trained and tested on the UN French-English parallel text corpus has higher accuracy than the Malay-English parallel text corpus because the training and testing data are from the same domain. However, the training and testing data in the Malay-English parallel text corpus are from different domains. Adding more training sentences improved the accuracy of all classifiers. The parallel LSTM+Attention+CNN classifier obtained classification accuracies of 81.96% and 87.01% with the Malay-English model and French-English model, respectively, using 10 thousand training sentences. When the number of sentences was 100 thousand, the accuracies of the proposed classifier increased to 90.57% and 96.12% when classifying Malay-English sentences and French-English sentences, respectively. The parallel bi-directional LSTM has the second-best accuracy overall. The parallel CNN classifier was slightly worse than parallel bi-directional LSTM in terms of accuracy. Based on the result, the CNN classifier requires more than 20 thousand parallel sentences to obtain a reasonably good classification accuracy. The feedforward neural network was the worst performer among all the classifiers.

The LSTM, bi-directional LSTM, and CNN in the classifiers encode given sentences to sentence vectors. Encoding sentences using CNN was first proposed by Kim (2014) to be used in sentiment classification. After the sentences are encoded in the parallel bi-directional LSTM and CNN classifier, the subsequent layers compare the encoded sentences in terms of their similarity and classify similar sentences as parallel and vice versa. In the parallel bi-directional LSTM classifier and the parallel CNN classifier, the higher layers were the same, consisting of two dense layers. Based on the results of the parallel bi-directional LSTM classifier and parallel CNN classifier, we observed that the bi-directional LSTM is slightly better in encoding sentences than CNN in this study, as bi-directional LSTM achieved higher accuracy in the classification tasks.

On the other hand, there is an attention mechanism in our proposed LSTM with attention and CNN classifier after the LSTMs encode the sentences to sentence vectors. The attention mechanism learns the relationship between the lexical items between the sentence vectors. Figure 10 shows an interesting visualization of the attention matrix when two test sentences, "*dia benci akan pergaduhan dan pembunuhan*" and "he abhors fighting and killing," were input to the classifier. The attention weights were normalized and set to brighter color for a higher value. In Figure 10, the attention cells that align between the words '*dia*' and 'he,' '*pergaduhan*' and 'fighting,' '*pembunuhan*' and 'killing' have high

values. During training, the attention mechanism learns to associate between lexical items of the given language pair that frequently appear together by assigning higher weights to them. Often, these are words that are a translation of each other. The attention mechanism is the main contributor to the increment in the classification accuracy because the attention learns the soft lexical alignments in the sentence. Subsequently, the CNN layer then learns the patterns of the attention matrix to classify it as parallel or non-parallel.

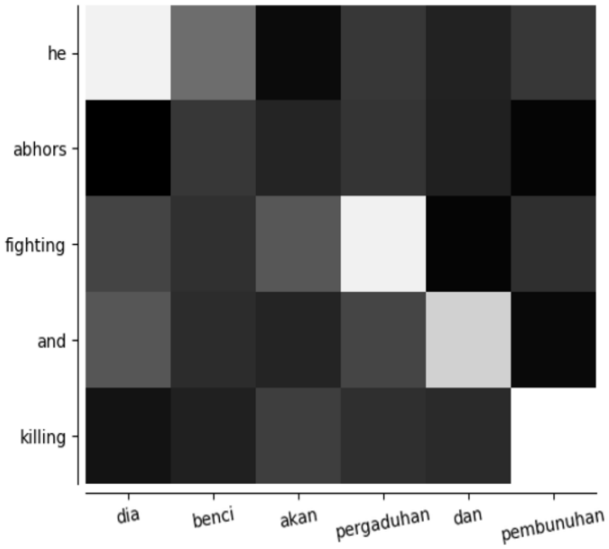

Figure 10. Visualizing the attention matrix

In the second experiment, we evaluated the alignment of Malay and English documents using the proposed sliding window approach with different classifiers. For this study, the classifiers' models that were trained using all the training data were used. The window size d was set to three for all the experiments, and the step size J was estimated using Equation 11. Besides the classifiers, we also tested BleuAlign in aligning the documents. BleuAlign needs an MT system to produce hypothetical translation for the source sentences before using sentence alignment. We used a statistical machine translation system (Yeong et al., 2019) for this purpose. Table 5 shows the accuracies of the sentence alignment using different classifiers and the BleuAlign system. In general, the result aligns with the performance of the classifiers, where the classifier with the higher accuracy in the first task achieved a higher sentence alignment accuracy. The proposed parallel LSTM+Attention+CNN classifier obtained the highest sentence alignment accuracy on the research and Classical Malay documents. The accuracies obtained for the two tasks were 95.75% and 51.24%, respectively. The second-best classifier was the parallel bi-directional LSTM, which scored 86.06% and 36.82%, respectively, for the two tasks.

On the other hand, the BleuAlign system achieved 78.18% and 23.01% accuracy for the two-sentence alignment tasks. From the experiment result, using the sliding window approach with the classifiers increased the accuracy of sentence alignments on the research document.

The sliding window limits the search of the sentences to within the search window (which is 2 x window size + 1). It improves the searching time and accuracy when aligning the research document due to the heuristic that sentences are aligning in monotonic order. However, without limiting the search space, the accuracy of the classifier in aligning the sentences may reduce due to the following three reasons. First, a sentence in a source document can be translated to zero or more sentences in the target document, or more than one sentence in the source document can be translated to a single target sentence. Second, when we evaluated the classifiers in the previous experiment, there were no partially matched sentences in the training or testing data since the training and testing data were generated from a parallel corpus. These partially match sentences may increase the error rate of the classifier since the classifier was not specifically trained for it.

Nevertheless, there was no drop in the accuracies in the experiment result when using the sliding window with the classifier, even though about 20% of the source sentences and target sentences were partially matched sentences. Second, when the search space increases, the probability for a classifier to make an error by matching a source sentence to similar target sentences will increase. For instance, consider two target sentences that differ only in one word. The possibility for the classifier to make an error will increase in this case. Third, the difference in testing and training domain may reduce the accuracy of the classifier in the alignment. For instance, in the case of French-English parallel sentence classification, the training and testing data were from the same domain. Thus, it achieved higher classification accuracy than the Malay-English parallel text classification where the training and testing data were acquired from different sources. Finally, the sliding window approach may reduce the effect of domain mismatch. It can be observed in the slight improvement in the accuracy of the sentence alignment when the Malay-English classifier was used with the sliding window. However, if the domain is very different, for instance, in the case of the alignment of sentences in classical Malay documents, the accuracy may also drop.

One of the challenges in aligning Classical Malay sentences is the differences in Classical Malay writing compared to Standard Malay. Besides, the English translation of the Classical Malay literature was written using semantic/free translation, increasing the alignment difficulty. Therefore, Classical Malay and Standard Malay are compared below, with example sentences from *Hikayat Hang Tuah*.

- The use of *'maka'* (then) and '*hatta*' as the start of the sentence. These words function as punctuation words in Jawi. Example: "*maka baginda pun tersenyum*" (His Majesty smile)

Table 5

Accuracy of different classifiers and BleuAlign in aligning sentences in Malay document and English document

Classifier/System	Classical Malay	Research Document
Parallel LSTM+Attention+CNN	51.24%	95.75%
Parallel Bidirectional LSTM	36.82%	86.06%
Parallel CNN	27.36%	82.42%
Feedforward	28.36%	58.79%
BlueAlign	23.01%	78.18%

- Passive sentence is frequent in Classical Malay text. Example: "*maka digelar oleh baginda bendahara paduka raja.*" (He was bestowed by His Majestry bendahara paduka raja.)
- Frequently used of the particle *'pun'* and *'lah'* in Classical Malay text. *'pun'* and *'lah'* form special feature structures in Classical Malay. The '*pun-lah*' structure indexes an event (*-lah*) and the participant (pun) who or which will be under investigation (Ajamiseba, 1983). Example: "*maka segala menteri pun kembalilah ke rumahnya.*"
- Classical Malay sentences are often long and convoluted. The longest test sentence is 80 words long!
- The words/phrases with the same surface form in Malay and Classical Malay have different meanings. For example, the phrase '*berapa lamanya ...*" means "after a while..." in Classical Malay but in Standard Malay, it means "how long...." The corresponding phrase in Standard Malay is "*selepas seketika*". Example: "*hatta berapa lamanya maka Tuan Puteri Kemala Ratna Pelinggam pun besarlah...*"

CONCLUSION

In this study, the parallel LSTM with attention and CNN is proposed for classifying two sentences as parallel/non-parallel sentences. The parallel LSTM+Attention+CNN classifier shows a higher classification accuracy than the feedforward classifier, bi-directional LSTM classifier, and CNN classifier. A sliding window that matches the source sentence and target sentences for alignment is also proposed for aligning translated documents. The sliding window selects sentence alignments with the highest overall probability and assumes sentence alignment appears in monotonic order. In aligning Malay-English sentences in the research document and Classical Malay-English document, the parallel LSTM+Attention+CNN approach produced better sentence alignment compared to the other baseline systems. Thus, the sliding window approach is suitable for the proposed classifier to construct a parallel text from translated documents. However, it is not suitable for comparable texts, for example, similar news in different languages by different

publishers or similar Wikipedia topics in different languages where the sentence alignments may appear in a free order, and the only fragment of a sentence may be aligned. Our future work will be to extend our study to this problem.

ACKNOWLEDGMENT

This research was supported by The Research University Grant (RUI) (1001. PHUMANITI.8016043) from Universiti Sains Malaysia.

REFERENCES

Ahmad, K. (2017). *Hikayat Hang Tuah* [The epic of Hang Tuah]. Dewan Bahasa & Pustaka.

Ajamiseba, D. C. (1983). *A classical malay text grammar: Insights into a non-wester text tradition*. Australian National University.

Almeman, K., Lee, M., & Almiman, A. A. (2013). Multi dialect Arabic speech parallel corpora. In *Proceedings International Conference on Communications, Signal Processing, and Their Applications (ICCSPA)* (pp. 1-6). IEEE Publishing. http://dx.doi.org/10.1109/ICCSPA.2013.6487288

Brown, P. F., Lai, J. C., & Mercer, R. L. (1991). Aligning sentences in parallel corpora. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics* (pp. 169-176). Berkeley. http://dx.doi.org/10.3115/981344.981366

Chen, S. F. (1993). Aligning sentences in bilingual corpora using lexical information. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics* (pp. 9-16). ACM Publishing. http://dx.doi.org/10.3115/981574.981576

Gale, W. A., & Church, K. W. (1993). A program for aligning sentences in bilingual corpora. *Computational Linguistics, 19*, 75-102. http://dx.doi.org/10.3115/981344.981367

Grégoire, F., & Langlais, P. (2017). *A deep neural network approach to parallel sentence extraction*. ArXiv preprint.

Khaw, J. Y. M., Tan, T. P., & Ranaivo, B. (2021). Kelantan and Sarawak Malay dialects: Parallel dialect text collection and alignment using hybrid distance-statistical-based phrase alignment algorithm. *Turkish Journal of Computer and Mathematics Education, 12*(3), 2163-2171. https://doi.org/10.17762/turcomat.v12i3.1160

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1746-1751). Association for Computational Linguistics Publishing. http://dx.doi.org/10.3115/v1/D14-1181

Lim, S. L. O., Lim, H. M., Tan, E. K., & Tan, T. P. (2020). Examining machine learning techniques in business news headline sentiment analysis. In R. Alfred, Y. Lim, H. Haviluddin & K. O. Chin (Eds.), *Computational Science and Technology* (pp. 363-372). Springer. https://doi.org/10.1007/978-981-15-0058-9_35

Luo, S., Ying, H., & Yu, S. (2021). *Sentence alignment with parallel documents helps biomedical machine translation*. ArXiv preprint.

Luong, M. T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In *Proceedings of Conference on Empirical Methods in Natural Language Processing* (pp. 1412-1421). Association for Computational Linguistics Publishing. http://dx.doi.org/10.18653/v1/D15-1166

Ma, X. (2006, May 22-28). Champollion: A robust parallel text sentence aligner. In *Proceedings of Fifth International Conference on Language Resources and Evaluation* (pp. 489-492). Genoa, Italy.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Processing Information Systems 26* (pp. 3136-3144). Curran Associates, Inc.

Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532-1543). Association for Computational Linguistics Publishing. http://dx.doi.org/10.3115/v1/D14-1162

Salleh, M. (2010) *The epic of Hang Tuah*. Malaysian Institute of Translation & Books.

Sennrich, R., & Volk, M. (2010, November 4 - December 14). MT-based sentence alignment for OCR-generated parallel texts. In *The Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)* (pp. 1-11). Denver, Colorado.

Stratos, K., Collins, M., & Hsu D. (2015). Model-based word embeddings from decompositions of count matrices. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics* (pp. 1282-1291). Association for Computational Linguistics Publishing. http://dx.doi.org/10.3115/v1/P15-1124

Vulić, I., & Moens, M. F. (2015). Bilingual word embeddings from non-parallel document-aligned data applied to bilingual lexicon induction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics* (Vol. 2, pp. 719-725). Association for Computational Linguistics Publishing. http://dx.doi.org/10.3115/v1/P15-2118

Wołk, K., & Marasek, K. (2014). A sentence meaning based alignment method for parallel text corpora preparation. *New Perspectives in Information Systems and Technologies, 1*, 229-237. http://dx.doi.org/10.1007/978-3-319-05951-8_22

Yeong, Y. M., Tan, T. P., & Gan, K. H. (2019) A hybrid of sentence-level approach and fragment-level approach of parallel text extraction from comparable text. *Procedia Computer Science, 161*, 406-414. http://dx.doi.org/10.1016/j.procs.2019.11.139

Zhou, P., Qi, Z., Zheng, S., Xu, J., Bao, H., & Xu, B. (2016, December 11-16). Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling. In *Proceedings of the 26th International Conference on Computational Linguistics* (pp. 3485-3495). Osaka, Japan.

Ziemski, M., Junczys-Dowmunt, M., & Pouliquen, B. (2016, May 23-28). The United Nations parallel corpus v1.0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016* (pp. 3530-3534). Portorož, Slovenia.