applied sciences

MDPI

*Article*

# Inter-Sentence Segmentation of YouTube Subtitles Using Long-Short Term Memory (LSTM)

**Hye-Jeong Song [1,2], Hong-Ki Kim [1,2], Jong-Dae Kim [1,2], Chan-Young Park [1,2] and Yu-Seop Kim [1,2,*]**

1 School of Software, Hallym University, Chuncheon-si, Gangwon-do 24252, Korea; hjsong@hallym.ac.kr (H.-J.S.); homg93@gmail.com (H.-K.K.); kimjd@hallym.ac.kr (J.-D.K.); cypark@hallym.ac.kr (C.-Y.P.)

2 Bio-IT Center, Hallym University, Chuncheon-si, Gangwon-do 24252, Korea

* Correspondence: yskim01@hallym.ac.kr; Tel.: +82-10-2901-7043

Received: 18 January 2019; Accepted: 8 April 2019; Published: 11 April 2019

check for updates

**Abstract:** Recently, with the development of Speech to Text, which converts voice to text, and machine translation, technologies for simultaneously translating the captions of video into other languages have been developed. Using this, YouTube, a video-sharing site, provides captions in many languages. Currently, the automatic caption system extracts voice data when uploading a video and provides a subtitle file converted into text. This method creates subtitles suitable for the running time. However, when extracting subtitles from video using Speech to Text, it is impossible to accurately translate the sentence because all sentences are generated without periods. Since the generated subtitles are separated by time units rather than sentence units, and are translated, it is very difficult to understand the translation result as a whole. In this paper, we propose a method to divide text into sentences and generate period marks to improve the accuracy of automatic translation of English subtitles. For this study, we use the 27,826 sentence subtitles provided by Stanford University's courses as data. Since this lecture video provides complete sentence caption data, it can be used as training data by transforming the subtitles into general YouTube-like caption data. We build a model with the training data using the LSTM-RNN (Long-Short Term Memory – Recurrent Neural Networks) and predict the position of the period mark, resulting in prediction accuracy of 70.84%. Our research will provide people with more accurate translations of subtitles. In addition, we expect that language barriers in online education will be more easily broken by achieving more accurate translations of numerous video lectures in English.

**Keywords:** speech to text; translate; period generation; subtitling

## 1. Introduction

Speech to Text (STT) [1,2] is a process in which a computer interprets a person's speech and then converts the contents into text. One of the most popular algorithms is HMM (Hidden Markov Model) [3], which constructs an acoustic model by statistically modeling voices spoken by various speakers [4] and constructs a language model using corpus [5].

Machine Translation (MT) is a sub-field of computational linguistics that investigates the use of software to translate text or speech from one language to another (https://en.wikipedia.org/wiki/Machine_translation). MT has been approached by rules [6], examples [7], and statistics [8]. Currently Neural Machine Translation (NMT) [9] has dramatically improved MT performance, and there are a lot of translation apps, such as iTranslate (https://www.itranslate.com/) and Google Translate (https://translate.google.com/), competing in the market.

A recent video-sharing site, YouTube's automated captioning system [10] combines STT with MT technology. When uploading a video, YouTube extracts the voice data and writes subtitle files, and translates the file into the desired language. These techniques are of great help to users who cannot speak in the language spoken in the content. However, if the speaker does not deliberately distinguish between sentences, for example if they do not make a pause between sentences, multiple sentences are recognized as a single sentence. This problem significantly degrades the performance of the MT. What's even worse is that YouTube manages subtitles by time slots, not by utterances. The subtitles automatically generated in the actual YouTube are divided into units of scene, and these are to be translated.

In this paper, we propose an automatic sentence segmentation method that automatically generates a period mark using deep neural networks to improve the accuracy of automatic translation of YouTube subtitles.

In natural language processing, a period is a very important factor in determining the meaning of a sentence. Table 1 shows sentences with different meanings according to the different period positions. The example sentence is an STT-generated sentence with periods and capital letters are removed. This sentence is divided into completely different sentences, such as Case 1 and Case 2, depending on the position of the period, and the generated sentences have different meanings. If the original text is divided as in Case 1, here "he" is "Sam John". On the other hand, if divided as in Case 2, "he" would be "John" and "John" should eat "Sam".

**Table 1.** An example of distinction (segmentation) ambiguity in STT-generated sentences without periods and capital letters.

| Example Sentence | I told him eat sam john called imp followed the command |
|---|---|
| Case 1 | I told him eat. Sam John called imp followed the command. |
| Case 2 | I told him eat Sam. John called imp followed the command. |

Studies on past real-time sentence boundary detection and period position prediction [11–13] have attempted to combine words and other features (pause duration, pitch, etc.) into one framework. A research [14] verified that the detection process could improve translation quality. In addition, a research has been conducted to automatically detect sentence boundaries based on a combination of N-gram language models and decision trees [15]. Currently, Siri, Android, and Google API (https://cloud.google.com/speech-to-text/docs/automatic-punctuation) insert punctuation marks after recognition. However, it is more important for them to distinguish whether the punctuation of a sentence is a period or a question mark, rather than finding and recognizing appropriate position when recognizing multiple sentences at once. Because these studies were focusing on speech recognition and translation, they relied on some acoustic features, which could not be provided by YouTube scripts. Chinese sentence segmentation [16] created a statistical model using data derived from Chinese Treebank, and predicted the positions of periods based on the model. This research was different from other mentioned research studies because it was using only text data.

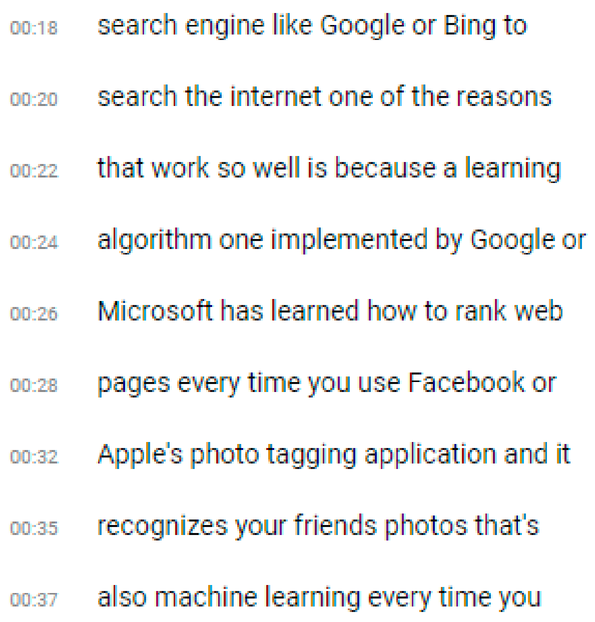
This paper presents a new sentence segmentation approach based on neural networks and YouTube scripts that is relatively less dependent on word order and sentence structure. We measures the performance of this approach. We used the 27,826 subtitles included in the online lectures provided by Stanford University for this study. These lecture videos provide well-separated subtitle data in sentence units. Therefore, this subtitles can be converted into the format of automatic subtitles provided by YouTube, and can be used as training data for the model to classify whether the period is present or not. We use Long-Short Term Memory (LSTM) [17] of Recurrent Neural Network (RNN) [18], which has excellent performance in natural language processing, to build a model with data and predict the position of the punctuation mark. LSTM showed potential to be adapted to punctuation restoration in speech transcripts [19], which combines textual features and pause duration. Although RNN has shown good performance with various input lengths, we sacrificed some of this benefits by making the data length similar to YouTube subtitles. In this study, we build the input as closely as possible to YouTube scripts, and try to locate the punctuation marks using only text features.

This paper is composed as follows. Section 2 describes the experimental data used in this study and their preprocessing process. Section 3 explains neural network-based machine learning and explains algorithms and models used in this study. In Section 4, the process of the period prediction and the sentence segmentation experiment are explained in detail. Finally, in Section 5, the present study is summarized and conclusions are presented.

## 2. Materials

### *2.1. Data*

In this paper, we collect 27,826 sentences from 11,039 sentences of "Natural Language Processing with Deep Learning" class and 16,787 sentences of "Human Behavioral Biology" class provided by Stanford University. The videos of these two lessons provide caption data of the complete sentence. Therefore, subtitles can be transformed into an automatic subtitle format provided by YouTube and used as training data to learn the position of a period. To do this, we convert the complete caption data into the format shown in Figure 1.



| | |
|---|---|
| 00:18 | search engine like Google or Bing to |
| 00:20 | search the internet one of the reasons |
| 00:22 | that work so well is because a learning |
| 00:24 | algorithm one implemented by Google or |
| 00:26 | Microsoft has learned how to rank web |
| 00:28 | pages every time you use Facebook or |
| 00:32 | Apple's photo tagging application and it |
| 00:35 | recognizes your friends photos that's |
| 00:37 | also machine learning every time you |

**Figure 1.** Examples of captions generated automatically in YouTube.

### *2.2. Preprocessing*

Figure 1 shows that company names such as Google (00:24), Microsoft (00:26), Facebook (00:28), and Apple (00:32) are written in capital letters. However, some people did not capitalize the first letters due to the nature of their own subtitles, so we converted the whole data to lower case. Also, if you do not do lowercase conversion, word embedding recognizes "Google" and "google" as different words.

Figure 2 is a flowchart showing the preprocessing of this study. First, the exclamation mark (!) and the question mark (?) are changed to a period (.) mark to indicate the end of the sentence. After that, sentences with less than 7 words ended by the period are excluded from the data, which means the minimum length required for learning. The eliminated sentences include sentences consisting mainly of simple greetings or nouns. Automatically generated subtitles do not have apostrophes (') or other punctuation. Therefore, only the period position of the actual subtitle is stored as a reference, and all remaining punctuation marks, such as commas (,), double quotation marks (" "), and single quotation marks (' '), are removed.

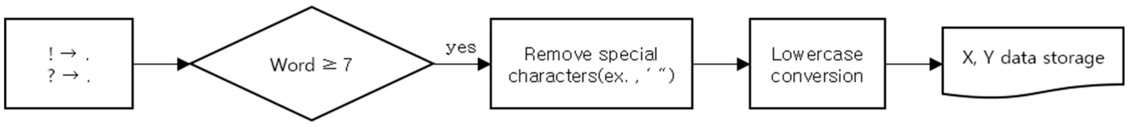


**Figure 2.** Data Preprocessing.

The preprocessed data is processed in the same way as Table 2 to learn long-short term memory (LSTM). First, given an original data sentence, a window of size 5 is created by moving one word at a time. In this study, only windows containing a period were considered as learning data. That is, in the first line of Table 2, "no one use them. And", "one use them. and so", "use them. and so the", "them. and so the cool", ". and so the cool thing" are generated as a candidate for the Ex. Only one of these candidates is selected as the ***Ex***.

**Table 2.** Examples of preprocessed data.

| | |
|---|---|
| **Raw Data** | but if you wanted to parse the web no one use them. **and so the cool thing** was that ... |
| Ex.1 | ['.', 'and', 'so', 'the', 'cool', 'thing'] |
| X_DATA[0] | ['and', 'so', 'the', 'cool', 'thing'] |
| Y_DATA[0] | [1, 0, 0, 0, 0] |
| **Raw Data** | and so the cool thing was that by doing this as neural network dependency parser we were able to **get much better accuracy. we** were able ... |
| Ex.2 | ['get', 'much', 'better', 'accuracy', '.', 'we'] |
| X_DATA[1] | ['get', 'much', 'better', 'accuracy', 'we'] |
| Y_DATA[1] | [0, 0, 0, 0, 1] |

The ***Ex*** is converted into X_DATA only with the included words again, and these words become the input of the learning. For learning, words are converted to vector representations using word embedding. Y_DATA, which is a reference of learning, shows the period to the left of a word. In Table 2, you can see that a period appears at the left of the first word. The second example in Table 2 shows that a period appears on the left of the fifth word. Through this process, LSTM is learned with X_DATA and Y_DATA, and a model that can predict the end of a sentence is created.

## 3. Methods

In this paper, we use LSTM of an artificial neural network among machine learning methods to predict whether or not to split the sentence. To do this, we create a dictionary with a number (index) for each word, express each word as a vector using word embedding, and concatenate 5 word vectors and use it as X_DATA.

Figure 3 shows how data are processed in the program. The five words in Figure 3 are taken from the second example in Table 2. First, all words get vector values of 100 dimensions through word embedding using Word2Vec. These vector values also contain information about the position of the period. Second, the words that have been converted to vectors enter the LSTM layer and this output is calculated while being affected by the previous output. Finally, these outputs predict that a period must be generated between the "accuracy" and "we" by softmax function.

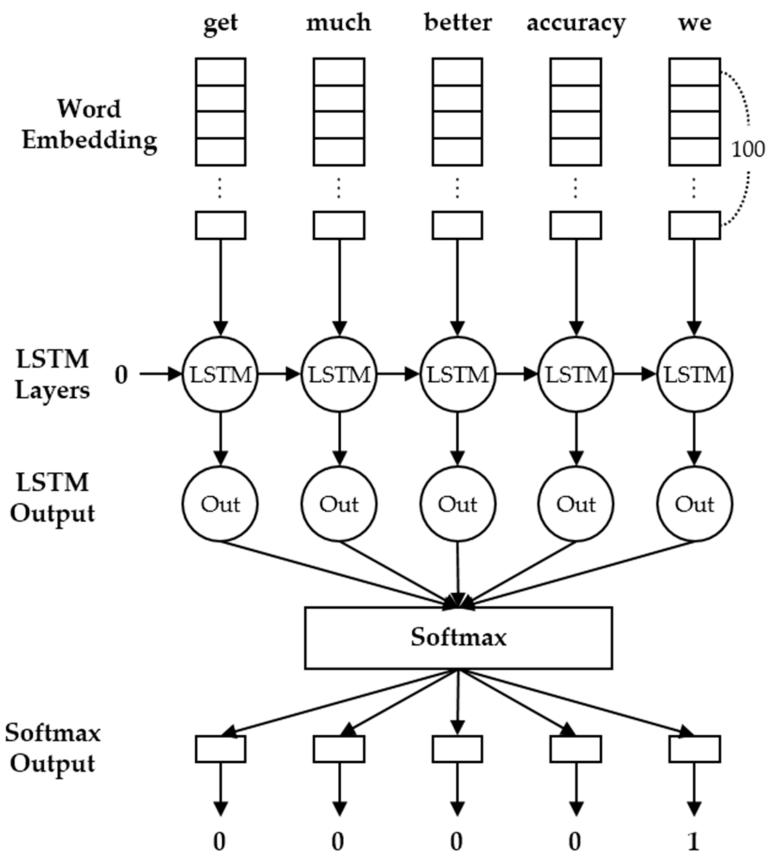


**Figure 3.** Schematic of the program to learn and generating period using Word2Vec and LSTM.

### *3.1. Word Embedding*

Word embedding is a method of expressing all the words in a given corpus in a vector space [20]. Word embedding allows us to estimate the similarity of words, which makes it possible to achieve higher performance in various natural language processing tasks. Typical word embedding models are Word2Vec [21–23], GloVe [24], and FastText [25]. In this paper, words are represented as vectors by using Word2Vec, which is most widely known to the public.

Word2Vec is a continuous word embedding learning model created by several researchers, including Google's Mikolov in 2013. The Word2Vec model has two learning models: Continuous Bag of Words (CBOW) and Skip-gram. CBOW basically creates a network for inferring a given word by using surrounding words as an input. This method is known to have good performance when data is small. On the other hand, the Skip-gram uses the given word as an input to infer the surrounding word. This method is known to show good performance when there is a lot of data (https://www.tensorflow.org/tutorials/representation/word2vec).

### *3.2. RNN (Recurrent Neural Network)*

RNN (Recurrent Neural Network) is a neural network model that is suitable for learning time-series data. In the previous neural network structure, it is assumed that the input and output are independent of each other. However, in RNN, the same activation function is applied to every element of one sequence, and the output result is affected by the previous output result. However, the actual implementation of RNN has a limitation in effectively handling only relatively short sequences. This is called vanishing gradient problem, and a long-short term memory (LSTM) has been proposed to overcome this problem.

As shown in Figure 4, LSTM solved the vanishing gradient problem by adding an input gate, a forget gate, and an output gate to each path that compute the data in the structure of the basic RNN.

These three gates determine what existing information in the cell will be discarded, whether to store new information, and what output value to generate.

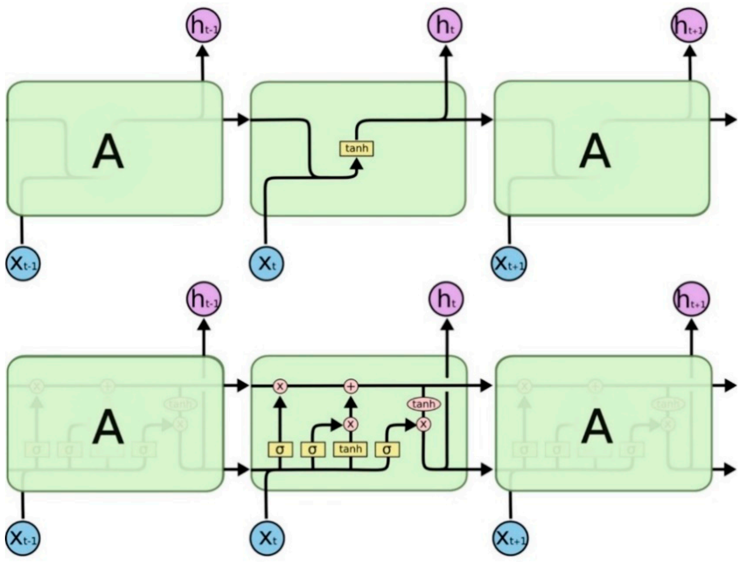


**Figure 4.** Unrolled RNN (Recurrent Neural Networks) cell structure (top) vs. LSTM (Long-Short Term Memory) cell structure (bottom) (http://colah.github.io/posts/2015-08-Understanding-LSTMs/).

LSTM has been successfully applied to many natural language processing problems. Especially, it shows good performance in the language modelling [26], which calculates the probability of seeing the next word in the sentence, and the machine translation field [27], which decides which sentence to output as the result of automatic translation. In this study, we use this characteristic of LSTM to train the features of sentence segmentation of a series of consecutive words. We also create a model that can automatically segment sentences by creating a degree of segmentation of the sentence as the output value of the network.

## 4. Experimental Results

The data of this paper is composed of 27,826 subtitles (X_DATA, Y_DATA) in total. Training data and test data were randomly divided into 7:3 ratios. Table 3 shows the Hyper parameters used in this study and the performance of the experiment.

**Table 3.** Hyper parameters and result performance.

| Data and Hyper Parameters | Figure |
|---|---|
| All data | 27,826 |
| Training data | 19,478 |
| Test data | 8348 |
| Embedding dimension | 100 |
| RNN cell layer | 3 |
| Epochs | 2000 |
| Learning rate | 0.115 |
| Cost | 0.181 |
| Accuracy | 70.84% |

As can be seen in Table 3, the words were represented as 100-dimensional vectors and training was repeated 2000 times with a training set. As a result, the final cost is 0.181 and the accuracy is 70.84%.

We created a confusion matrix of Table 4 to evaluate our method in detail. In this table, A, B, C, D, and E correspond to classes of Y_DATA in Table 2, in which A is “1, 0, 0, 0, 0”, B is “0, 1, 0, 0, 0”, and so on. We excluded the prediction case of "0, 0, 0, 0, 0", since this case could be the further addressed in the future.

**Table 4.** Confusion matrix of 5 classes' prediction.

| | | Predicted | | | | |
|---|---|---|---|---|---|---|
| | | A | B | C | D | E |
| **Actual** | A | 1066 | 40 | 49 | 88 | 99 |
| | B | 29 | 1278 | 43 | 49 | 63 |
| | C | 75 | 28 | 1276 | 52 | 45 |
| | D | 101 | 79 | 46 | 1206 | 48 |
| | E | 129 | 112 | 85 | 89 | 1088 |

Table 5 shows precision, recall, and f-measure values of each class. The average f-measure is over 80%, which is a relatively higher performance than previous research studies [11–15], even though we did not use acoustic features. Five classes did not show distinguished difference in their prediction performances. We did not consider which word could be a candidate of segmentation because the process also requires additional calculation and resources. We did not use any manual annotation work compared to other works proposed earlier. Given those considerations, the performance shows much potentials.

**Table 5.** Precision, Recall, and *f*-measure values of each class.

| | **A** | **B** | **C** | **D** | **E** | **Average** |
|---|---|---|---|---|---|---|
| **Precision** | 0.761429 | 0.83149 | 0.851234 | 0.818182 | 0.810127 | 0.814492 |
| **Recall** | 0.83249 | 0.880165 | 0.864499 | 0.814865 | 0.723886 | 0.81555 |
| ***f*-measure** | 0.777535 | 0.855135 | 0.857815 | 0.81652 | 0.764582 | 0.814317 |

Figure 5 shows the cross validation data and training data cost for learning epochs. By checking the cross validation, it is possible to determine the optimum number of epochs and check whether there is a problem of overfitting or underfitting. In this graph, after 1000 repetitions, the CV graph and the training graph are slightly different. Therefore, it can be concluded that there is no significant meaning for the more iterative learning.

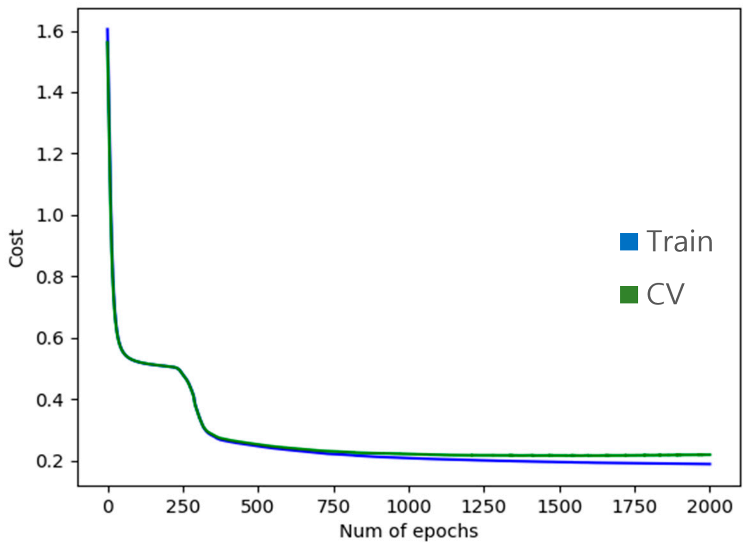


**Figure 5.** Graph of Training cost and Cross Validation (CV) cost according to learning rate.

Table 6 shows the predicted test data based on the learned model. A data shows correct prediction and B data shows wrong prediction.

**Table 6.** Examples of correct prediction and wrong prediction.

| **A Data** | **['a' 'psychiatric' 'disease' 'the' 'notion']** |
|---|---|
| answer | [0 0 0 1 0] |
| prediction | [0 0 0 1 0] |
| predicted period | A psychiatric disease. The notion |
| **B Data** | **['and' 'all' 'of' 'that' 'something']** |
| answer | [0 0 0 0 1] |
| correctly predicted period | and all of that. Something |
| prediction | [1 0 0 0 0] |
| wrongly predicted period | And all of that something |

## 5. Conclusions

In this paper, we present a method to find proper positions of period marks in YouTube subscripts, which can help improve the accuracy of automatic translation. This method is based on Word2Vec and LSTM. We cut off the data to make them similar to YouTube subscripts. We tried to predict whether a period can come between each word or not. In the experiment, the accuracy of the approach was measured to be 70.84%.

In a future study, we will apply other neural net models, such as CNN, attention, and BERT, to bigger data to improve the accuracy. To do this, we need to collect subscript data from a wider range of online education sites. Second, we want to collect more YouTube subtitle data in various areas and translate it into Korean to build a parallel corpus. We will develop the software tools needed to build the corpus. Finally, we will develop a speech recognition tool based on the open API to create the actual YouTube subtitle files. With those steps, we will be able to build all the necessary processes from the YouTube voice to the Korean subtitles creation.

**Author Contributions:** H.-J.S. provided Neural Network fundamentals and implementation. H.-K.K. mainly built this research and wrote this paper. J.-D.K. and C.-Y.P. advised about Speech-to-Text and machine translation. Y.-S.K. is a supervisor of this research.

**Funding:** This work was funded by the National Research Foundation of Korea Grant funded by the Korean Government (NRF-2017M3C4A7068188) and by Basic Science Research Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Science, Information and Communication Technology, and Future Planning (2015R1A2A2A01007333).

**Acknowledgments:** Byoung-Doo Oh gave his technical support for Deep Neural Networks and Tak-Seong Heo helped this research by implementing a part of the model.

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

## References

1. Bijl, D.; Henry, H.-T. Speech to Text Conversion. U.S. Patent No. 6,173,259, 9 January 2001.
2. Manning, C.D.; Christopher, D.M.; Hinrich, S. *Foundations of Statistical Natural Language Processing*, 1st ed.; MIT Press: Cambridge, MA, USA, 1999.
3. Krogh, A.; Larsson, B.; von Heijne, G.; Sonnhammer, E.L. Predicting transmembrane protein topology with a hidden Markov model: Application to complete genomes. *J. Mol. Biol.* **2001**, *305*, 567–580. [CrossRef] [PubMed]
4. Liao, H.; Erik, M.; Andrew, S. Large scale deep neural network acoustic modeling with semi-supervised training data for YouTube video transcription. In Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU 2013), Olomouc, Czech Republic, 8–12 December 2013.

5. Robinson, T.; Fransen, J.; Pye, D.; Foote, J.; Renals, S. WSJCAMO: A British English speech corpus for large vocabulary continuous speech recognition. In Proceedings of the International Conference on Acoustics, Speech, and Signal Processing ICASSP-95, Detroit, MI, USA, 9–12 May 1995; Volume 1.
6. Forcada, M.L.; Ginesti-Rosell, M.; Nordfalk, J.; O'Regan, J.; Ortiz-Rojas, S.; Perez-Ortiz, J.A.; Sanchez-Martinez, F.; Ramirez-Sanchez, G.; Tyers, F.M. Apertium: A free/open-source platform for rule-based machine translation. *Mach. Transl.* **2011**, *25*, 127–144. [CrossRef]
7. Sommers, H. Example-based machine translation. *Mach. Transl.* **1999**, *14*, 113–157. [CrossRef]
8. Brown, P.F.; Della Pieta, V.J.; Della Pietra, S.A.; Mercer, R.L. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Comput. Linguist.* **1993**, *19*, 263–311.
9. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. In Proceedings of the ICLR, San Diego, CA, USA, 7–9 May 2015.
10. Miyamoto, K.; Noriko, N.; Kenichi, A. Subtitle Generation and Retrieval Combining Document with Speech Recognition. U.S. Patent No. 7,739,116, 15 June 2010.
11. Di Francesco, R.J. Real-time speech segmentation using pitch and convexity jump models: Application to variable rate speech coding. *IEEE Trans. Acoust. Speech Signal Process.* **1990**, *38*, 741–748. [CrossRef]
12. Chen, C.J. Speech recognition with automatic punctuation. In Proceedings of the EUROSPEECH, Budapest, Hungary, 5–9 September 1999.
13. Batista, F.; Fernando, C.; Diamantino, M.; Nuno, J.; Trancoso, I. Recovering punctuation marks for automatic speech recognition. In Proceedings of the INTERSPEECH, Antwerp, Belgium, 27–31 August 2007.
14. Matusov, E.; Arne, M.; Hermann, N. Automatic sentence segmentation and punctuation prediction for spoken language translation. In Proceedings of the International Workshop on Spoken Language Translation (IWSLT), Kyoto, Japan, 27–28 November 2006.
15. Stolcke, A.; Shriberg, E.; Bates, R.A.; Ostendorf, M. Automatic detection of sentence boundaries and disfluencies based on recognized words. In Proceedings of the Fifth International Conference on Spoken Language Processing, Sydney, Australia, 30 November–4 December 1998.
16. Xue, N.; Yaqin, Y. Chinese sentence segmentation as comma classification. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Portland, Oregon, 19–24 June 2011; Short papers-Volume 2.
17. Hochreiter, S.; Jürgen, S. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
18. Mikolov, T.; Karafiát, M.; Burget, L.; Cernocký, J.; Khudanpur, S. Recurrent neural network based language model. In Proceedings of the Eleventh Annual Conference of the International Speech Communication Association, Makuhari, Japan, 26–30 September 2010.
19. Tilk, O.; Alum, T. LSTM for punctuation restoration in speech transcripts. In Proceedings of the INTERSPEECH, Dresden, Germany, 6–10 September 2015.
20. Levy, O.; Yoav, G. Neural word embedding as implicit matrix factorization. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014.
21. Mikolov, T.; Yih, W.T.; Zweig, G. Linguistic regularities in continuous space word representations. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Atlanta, GA, USA, 9–14 June 2013; pp. 746–751.
22. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. In Proceedings of the Advances in Neural Information Processing Systems, Stateline, NV, USA, 5–10 December 2013; pp. 3111–3119.
23. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
24. Pennington, J.; Socher, R.; Manning, C. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 26–28 October 2014; pp. 1532–1543.
25. Bojanowski P, Grave E, Joulin A, Mikolov T, Enriching word vectors with subword information. *arXiv* **2016**, arXiv:1607.04606.
26. Christopher Olah's Github Blog. Available online: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (accessed on 27 August 2015).

27. Sutskever, I.; Oriol, V.; Quoc, V.L. Sequence to sequence learning with neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014.