The Influence of Loss Function Usage at SIAMESE Network in Measuring Text Similarity

Suprapto¹, Joseph A. Polela²
Department of Computer Science and Electronics
Universitas Gadjah Mada
Yogyakarta, Indonesia

Abstract—In a text matching similarity task, a model takes two sequence of text as an input and predicts a category or scale value to show their relationship. A developed model is to measure the similarity - one of relationship between those two text. The model is SIAMESE network that implement two copies of same network of CNN, it takes text 1 and text 2 as the inputs respectively for two CNN networks. The output of each CNN network is features vector of the corresponding text input, both outputs are then fed by a loss function to calculate the value of loss (i.e. similarity). This research implemented two types of loss functions, i.e. Triplet loss and Contrastive loss. The usage purpose of these two types of loss functions was to see the influence toward the measurement results of similarity between two text being compared. The metrics used for this comparison are precision, recall, and F1-score. Based on the experimental results done on 1500 pairs of sentences, and varied on the epoch value starting from 10 until 200 with an increment of 10, showed the best result was for epoch value of 180 with precision 0.8004, recall 0.6780, and F1-score 0.6713 for Triplet loss function; and epoch value of 160 with precision 0.6463, recall 0.6440, and F1-score 0.6451 for Contrastive loss function gave the best performance. So that, the Triplet loss function gave better influence than Contrastive loss function in measuring similarity between two given sentences.

Keywords—Sentence; similarity; triplet; contrastive; CNN; Siamese; dataframe

I. INTRODUCTION

The very fast growth of information nowadays causes a particular problem, such as an overwhelming of information [21]. It is very likely among those collections of huge of information found some similar ones, so that, they can be grouped into several classes based on their similarity. In order to overcome the overwhelming of information, each class will only be represented by a single information. Obviously, to identify the similar information requires a process called similarity measurement. A text similarity measurements is one of text mining approach that capable of coping with the information overwhelming. This process begins with finding similar word for sentence, then paragraph, and finally document [6]. Text similarity approach will ease people to find relevance information. It has a great support in successness for text mining operations such as, searching and information retrieval (IR), text classification, information extraction (IE), document clustering [8], sentiment analysis [4] [10] [16][3] [13], machine translation, text summarization, and natural language processing (NLP). Text similarity measurement may be done by comparing text - text matching. In text comparison tasks, a model takes two texts as inputs and predicts a category or a scale value indicates the relationship between those two texts. A big number of varieties of tasks such as, natural language inference [2] [11], paraphrase identification [17], answer selection [19] could be consider as special form of text matching problems. Recently, deep neural network is the most popular choice for text mining. Semantic alignment and comparison of two sequence of texts are the key of neural text matching. Most of previous deep neural network contain single inter-sequence alignment layer. In order to make the alignment process fully used, model must take many external syntactical features or alignment as additional inputs at alignment layer [5] [7], adopt a complex alignment mechanism [17], or build a big number of post-process layers to analyze alignment results [7].

This research proposed two models to compute similarity value between two texts using SIAMESE network in which each uses two different types of loss function - Triplet loss and Contrastive loss. The model consists of two copies of same CNN networks, where each receives text (or sentence) as input. Subsequently, each CNN network results feature vector of each received text, and finally fed by loss function to compute the similarity value. Each model will be tested using the same dataset Quora Question Pair similarity taken from https://www.kaggle.com/c/quora-question-pairs. In order to see the influence of using these two types of loss function, the three metrics: precision, recall, and F1-score will be computed.

II. RELATED WORKS

Deep neural network is very dominant in text matching area. While semantic alignment and comparison between two text sequences is the core of text matching [17]. The very beginning task explores encoding of each sequence individually into a vector and then builds a neural network classifier over the two vectors. In this paradigm, recurrence [2], recursive [12], convolutional network [20] were used as sequence encoder. In this model, where encoding from a sequence independent with other sequence caused the last classifier had difficulty in modeling complex relations. Therefore, the subsequent tasks adopt framework matching aggregation to match two sequences at the lower level and aggregate results based on attention mechanism. DecomAtt used a simple form from attention for alignment and aggregate representations aligned by feed-forward network [15]. ESIM used a similar attention mechanism and implement bidirectional LSTMs as encoders and aggregators [5]. In order to improve model’s performance, the researcher adopted three main paradigms. The first paradigm used syntax that richer hand-writing features. HIM used syntactic parse tree [5]. The many usages of POS were found in previous tasks, some of them in [12]...
A. Learning Model

Based on the objective of the research, the model was built with three main components: twin network, similarity function, and output layer.

1) **Twin Network**: most feature’s extraction for text take place in this network. It has two copies of same networks, two networks share set of same weights. It capable of receiving two different inputs - two sequence of text. Each network in it is a convolutional network is comprised of three instances of the same feedforward network (with shared parameters). When fed with three samples, the network outputs two intermediate values - the L2 distances between the embedded representation of two of its inputs from the representation of the third. The three inputs will be denoted as $x$, $x^+$ and $x^-$, and the embedded representation of the network as $Net(x)$. In words, this encodes the pair of distances between each of $x^+$ and $x^-$, against the reference $x$, i.e., $\|Net(x) - Net(x^-)\|_2$ and $\|Net(x) - Net(x^+)\|_2$. The distance between reference input $x$ and positive input $x^+$ was minimized, on the other hand, the distance between reference input $x$ and negative input $x^-$ was maximized. The structure of triplet network is shown in Fig. 2 [9].

b. **Contrastive Network**

The contrastive loss function takes output from network for positive sample and computes the distance to an example from the same class and contrast it with distance to negative examples. In other word, the value of loss is low if positive samples are encoded to similar representations (closer), and negative examples are encoded to different representations (farer). The formula used to compute distance was cosine similarity distance as shown in equation (2).

$$Loss(x_p, x_q, y) = y \ast ||x_p - x_q||^2 + (1 - y) \ast \max(0, m^2 - ||x_p - x_q||^2)$$

(2)
IV. RESULTS AND DISCUSSION

As previously mentioned, the dataset used in the research was public. It was taken from https://www.kaggle.com/c/quora-question-pairs. The dataset was in CSV file containing 5000 pairs of Quora questions. The dataset was read to generate dataframe, and then the generated dataframe was divided into two: 70% for training and the other 30% for testing. The next process was building Word2Vec model using Gensim for embedding layer of deep learning. Then define function to transform sentence into vector containing index of Word2Vec’s vocabularies. Another two important functions to be defined were triplet and contrastive loss, continued to define the CNN Siamese network with both triplet and contrastive loss functions, and then trained it.

A. Dataframe

The dataframe was generated in several steps: first applying simple process both to question 1 (sent_1) and question 2 (sent_2), to do this there was a function simple_process taken from https://radimrehurek.com/gensim/utils.html #gensim.utils.simple_preprocess for tokenization. This process was done for all pairs of sentences in the dataset. From these all tokenized pairs of sentence, the number of tokens from the longest sentence was calculated based on the mean and deviation standard. This value was used to do padding, to make all tokenized pairs of sentences had the same length. In order to train the Siamese CNN model with triplet loss function, a negative sentence (the third tokenized sentence) must have been added to each pair of tokenized sentence (i.e. tokenized_sent_1 and tokenized_sent_2) for training dataframe. The format of training dataframe is shown in Fig. 3.

However, in the testing process for Siamese CNN model even with triplet loss function only need two tokenized sentences (tokenized_sent_1 and tokenized_sent_2). The dataframe format for testing was shown in Fig. 4.

B. Word2Vec Model

Before building the CNN Siamese network, the Word2Vec model was built using Gensim for embedding layer in deep learning. It was built with CBOW (https://iksinc.online/tag/continuous-bag-of-words-cbow/) with 20 iterations, vector length of 100, and window size was 5. The Word2Vec was trained once using string of words (tokenized sentence as a result of concatenation between tokenized of sentence 1 and tokenized of sentence 2), and saved it. During the process of training the model, there was a function required to convert a sentence into a vector containing Word2Vec vocabulary indices. For the model using triplet loss function, the converter changed three tokenized sentences (i.e. tokenized_sent_1, tokenized_sent_2, tokenized_sent_3) into three vectors containing indices of Word2Vec vocabularies (i.e. sent_1_ids, sent_2_ids, sent_3_ids), and two tokenized sentences (i.e. tokenized_sent_1, tokenized_sent_2) into two vectors containing indices of Word2Vec vocabulary (i.e. sent_1_ids, sent_2_ids) respectively in training and testing processes. However, for the model using contrastive loss function, the converter changed two tokenized sentences (i.e. tokenized_sent_1, tokenized_sent_2) into two vectors containing indices of Word2Vec vocabularies (i.e. sent_1_ids, sent_2_ids) in both training and testing processes. The sample of conversion results was shown in Fig. 5.

While the definition of the function that convert list of tokens to list of indices was shown in Fig. 6.

C. Triplet Loss

The Triplet loss function takes three inputs: baseline (an anchor sentence), positive (true sentence - one closest to an anchor), and negative (false sentence - one farthest to an anchor). Therefore, the objective of this function is to minimize the
D. Contrastive Loss

Unlike the triplet, the contrastive loss function only takes two inputs: positive sample and negative example. The objective is to contrast the distance between positive sample and example from the same class with the distance between positive sample and negative example.

The implementation of the triplet function was shown in Fig. 7.

```python
class TripletLoss(nn.Module):
    def __init__(self, margin=1.0):
        super(TripletLoss, self).__init__()
        self.margin = margin

    def forward(self, output1, output2, output3):
        loss = torch.pow(1 - F.pairwise_distance(output1, output2), 2) +
               torch.pow(F.pairwise_distance(output1, output3) - self.margin, 2)
        return loss.sum()
```

E. CNN Siamese Network

The CNN Siamese network was built with the following specifications:

1) The embedding layer was the Word2Vec model that had already been trained.
2) Three 2-dimensional convolutional layers with configurable hyperparameters.
3) For every convolutional layer used tanh as an activation function, dropout, and maxpool layer.

The architecture of CNN that was implemented in the CNN Siamese or just Siamese model was shown in Fig. 9 [22].

```python
---
# CNN Siamese network
---
```

The hyperparameter of the CNN Siamese network was shown in Fig. 10.

```python
---
# CNN Siamese's Hyperparameter
---
```

These two functions were only used during the training process to update the weights of network in order to converge; the triplet loss used two distances, and the contrastive loss only used one distance. While during the testing, no loss function needed, but the distance between two sentences. Because the model simply used the final weights resulted from training process.

The CNN Siamese model with triplet loss function consisting of three exact same models of CNN was trained using triple tokenized sentences (i.e. tokenized_sent_1, tokenized_sent_2, and tokenized_sent_3) from the dataframe training. The output of each CNN model was fed by Triplet loss function to calculate the loss value. On the other hand, the CNN Siamese model with contrastive loss function consisting of two exact same models of CNN was trained...
Using pair of tokenized sentences (i.e., tokenized sent 1, and tokenized sent 2) from the dataframe training. The output of each CNN model was fed by contrastive loss function to calculate the loss value. Finally, both model of CNN Siamese with triplet and contrastive loss function respectively were tested by feeding two tokenized sentences to be calculated their similarity value. Both testing were conducted using the same number of 1500 pairs of sentences (Quora questions), and varied for the values of EPOCH starting from 10 until 200 with an increment of 10 to obtain the values of three metrics: precision, recall and F1-score. The values of three matrices for each CNN Siamese model were shown respectively in Table I and Table II.

### Table I. The Precisions, Recalls and F1-Scores of the CNN Siamese with Triplet

| Epoch | Training Loss | Precision | Recall | F1-Score |
|-------|---------------|-----------|--------|----------|
| 10    | 64.3868       | 0.3762    | 0.6133 | 0.4663   |
| 20    | 19.4053       | 0.3762    | 0.6133 | 0.4663   |
| 30    | 8.6268        | 0.6277    | 0.6187 | 0.4896   |
| 40    | 4.9023        | 0.6469    | 0.6444 | 0.5513   |
| 50    | 2.2373        | 0.6762    | 0.6827 | 0.6602   |
| 60    | 2.3511        | 0.7073    | 0.7127 | 0.7074   |
| 70    | 1.8006        | 0.7254    | 0.7207 | 0.7224   |
| 80    | 1.4510        | 0.7433    | 0.7253 | 0.7288   |
| 90    | 1.2097        | 0.7526    | 0.7160 | 0.7194   |
| 100   | 1.0438        | 0.7520    | 0.6980 | 0.7003   |
| 110   | 0.9170        | 0.7562    | 0.6873 | 0.6880   |
| 120   | 0.8245        | 0.7746    | 0.6960 | 0.6956   |
| 130   | 0.7499        | 0.7774    | 0.6860 | 0.6838   |
| 140   | 0.6984        | 0.7834    | 0.6807 | 0.6768   |
| 150   | 0.6398        | 0.7892    | 0.6800 | 0.6752   |
| 160   | 0.6074        | 0.7885    | 0.6707 | 0.6642   |
| 170   | 0.5824        | 0.7907    | 0.6707 | 0.6639   |
| 180   | 0.5569        | 0.8004    | 0.6780 | 0.6713   |
| 190   | 0.5494        | 0.7973    | 0.6700 | 0.6621   |
| 200   | 0.5203        | 0.7986    | 0.6733 | 0.6659   |

### Table II. The Precisions, Recalls and F1-Scores of the CNN Siamese with Contrastive

| Epoch | Training Loss | Precision | Recall | F1-Score |
|-------|---------------|-----------|--------|----------|
| 10    | 16.2760       | 0.3762    | 0.6133 | 0.4663   |
| 20    | 5.8481        | 0.3762    | 0.6133 | 0.4663   |
| 30    | 2.6012        | 0.3760    | 0.6127 | 0.4660   |
| 40    | 1.4245        | 0.6290    | 0.6193 | 0.4921   |
| 50    | 0.8705        | 0.6370    | 0.6333 | 0.5434   |
| 60    | 0.5853        | 0.5921    | 0.6200 | 0.5508   |
| 70    | 0.4222        | 0.6068    | 0.6287 | 0.5824   |
| 80    | 0.3151        | 0.6113    | 0.6313 | 0.5976   |
| 90    | 0.2358        | 0.6071    | 0.6260 | 0.6030   |
| 100   | 0.1795        | 0.6199    | 0.6347 | 0.6190   |
| 110   | 0.1423        | 0.6246    | 0.6367 | 0.6258   |
| 120   | 0.1149        | 0.6283    | 0.6367 | 0.6305   |
| 130   | 0.0934        | 0.6332    | 0.6400 | 0.6354   |
| 140   | 0.0775        | 0.6331    | 0.6380 | 0.6384   |
| 150   | 0.0653        | 0.6225    | 0.6217 | 0.6244   |
| 160   | 0.0566        | 0.6463    | 0.6440 | 0.6451   |
| 170   | 0.0498        | 0.6336    | 0.6300 | 0.6315   |
| 180   | 0.0449        | 0.6436    | 0.6387 | 0.6407   |
| 190   | 0.0404        | 0.6299    | 0.6227 | 0.6254   |
| 200   | 0.0370        | 0.6440    | 0.6353 | 0.6384   |

In accordance with the testing results, it seemed that an ordinary CNN did not fit enough for this dataset even though each trained epoch showed its convergence, but the validation results were not good. This was caused either by the difference between distribution of words in the training data and the one in testing data or an ordinary CNN model found difficulty to distinguish two sentences with very similar structures, but in fact they were different semantically. For instance, “What’s your favorite political bumper sticker?” with “What was your favorite bumper sticker in the 2000s?”. These two sentences was tagged “not similar”, even their structures were similar, and there were some others. Conceptually, a CNN usually performs convolution with kernel. The kernel size used in this model was [1, 3, 5] multiplied by the size of embedding yielded by Word2Vec. If there are similar sentence structures, the result of the convolution will only be different in some elements of convolution’s results matrix. So that, the similarity between matrices of convolutions’s results from two sentence inputs will be high (i.e., two sentences are similar). Even though dropout and regularization had been implemented, this problem would still happened, because there were pair of sentences with similar structures would be considered either similar or not similar. In order to solve the problem it needs a language model and a more complex network. In this experiment, Word2Vec yielded vector of word based on surrounding words. It was not contextual, it means that one word would be represented by a vector that always the same.

From both Table I and Table II was derived chart comparing the two loss functions (triplet and contrastive) each for metric precision, recall and F1-score. Fig. 11, 12 and 13 show the chart of comparing metric between two loss functions, respectively for precision, recall and F1-score.

![Fig. 11. The Chart of Epoch versus Precision for Triplet and Contrastive Loss Function](image1)

![Fig. 12. The Chart of Epoch versus Recall for Triplet and Contrastive Loss Function](image2)

The three metric, precision, recall and F1-score were
computed using the formula derived from a confusion matrix.

V. CONCLUSIONS

The two CNN Siamese models had been successfully built, one with triplet loss function and the other with contrastive loss functions. The size of dataset used for training and testing were 3500 and 1500 respectively for the total of 5000. Both models were treated equally either in training or testing processes. In the training process, the model with triplet loss function consisted of three exact the same CNN models, while one with contrastive loss function only consisted of two exact the same CNN models. From the results of the model testing, it was seen that among 20 values of epoch there was one with epoch value of 180 with precision 0.8004, recall 0.6780, and F1-score 0.6713 for triplet loss function; and epoch value of 160 with precision 0.6463, recall 0.6440, and F1-score 0.6451 for contrastive loss function gave the best performance. So that, the triplet loss function gave better influence than contrastive loss function in measuring similarity between two given sentences.

REFERENCES

[1] Bird, S., at.al, 2009, Natural Language Processing with Python, Published by O'Reilly Media, Inc., 105 Gravenstein Highway North, Sebastopol, CA 95472.
[2] Bowman, S. R., Angeli, G., Potts, C. and Manning, C. D., 2015, "A large annotated corpus for learning natural language inference”. In Proceeding of the 2015 on Empirical Methods in Natural Language Processing, pages 632 - 642, Lisbon, Portugal. Association for Computational Linguistics.
[3] Bravo-Marquez, F., Mendoza, M., and Poblete, B., Meta-level models for big social data analysis, Knowl-Based Syst. 69 (2014) 86–99, http://dx.doi.org/10.1016/j.knosys.2014.05.016.
[4] Chauhan, V. K., Bansal, A. and Goel, A., Twitter Sentiment Analysis Using Vader, International Journal of Advance Research, Ideas and Innovations in Technology, vol. 4, 1 (2018), 485-489.
[5] Chen, Q., Zhu, X., Ling, Z. H., Wei, S., Jiang, H. and Inkpen, D., 2017, “Enhanced LSTM for Natural Language Inference”. In Proceedings of The 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages: 1657 - 1668; Vancouver, Canada. Association for Computational Linguistics.
[6] Gomma, W. H. and Fahmy, A. A., “A Survey of Text Similarity Approaches,” Int. J. Comput. Appl., vol. 68, no. 13, 2013, doi: https://doi.org/10.5120/11638 - 7118.
[7] Gong, Y., Luo, H. and Zhang, J., 2018, "Natural Language Inference over Interaction Space”, In Proceedings of the 6th International Conference on Learning Representations.
[8] Hidayat, E. Y., Firdausillah, F., Hastuti, K., Dewi, I. N. and Azhari, A., “Automatic Text Summarization Using Latent Dirichlet Allocation (LDA) for Document Clustering,” Int. J. Adv. Intell. Informatics, vol. 1, no. 3, p. 132, Dec. 2015, doi: https://doi.org/10.26555/ijain.v1i3.43.
[9] Hoffer, E., and Ailon, N., "Deep Metric Learning using Triplet Network" Accepted as a workshop contribution at ICLR 2015.
[10] Hutto, C. J. and Gilbert, E., VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Copyright © 2014, Association for the Advancement of Artificial Intelli-gence (www.aaai.org).
[11] Khot, T., Sabharwal and Clark, P., 2018, SciTail: A Textual Entailment Dataset from Science Question Answering. In Proceedings of AAAI.
[12] Kim, S., Hong, J. H., Kang, I. and Kwak, N., 2018, "Semantic Sentence Matching with Densely-connected Recurrent and Co-attentive Information". Computing Research Repository, arXiv: 1805.11360. Version 2.
[13] Liu, B.and Zhang, L., A survey of opinion mining and senti-ment analysis, in: Mining Text Data, Springer, 2012, pp. 415–463, http://dx.doi.org/10.1007/978-1- 4614-3223-4_13.
[14] Liu, X., Duh, K. and Gao, J., 2018, "Stochastic Answer Networks for Natural Language Inference". Computing Research Repository, arXiv: 1804.07888.
[15] Parikh, A., T., Tackstrom, O., Das, D. and Uszkoreit, J., 2016, "A Decomposable Attention Model for Natural Language Inference". In Proceedings of the 2016 Conference of Empirical Methods in Natural Language Processing (EMNLP), pages 1532 - 1543, Doha, Qatar. Association for Computational Linguistics.
[16] Ravi, K., and Ravi, V., A survey on opinion mining and sentiment analysis: Tasks, approaches and applications, Knowl-Based Syst. 89 (2015) 14–46, http://dx.doi.org/10.1016/j.knosys.2015.06.015.
[17] Wang, S. and Jiang, J., 2017, "A Compare-Aggregate Model for Matching Text Sequences”. In Proceedings of the 5th International Conference on Learning Representations.
[18] Yang, R., Zhang, J., Gao, X., Ji, F. and Chen, H., 2019, "Simple and Effective Text Matching with Richer Alignment Features". In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4699 - 4709, Florence, Italy, July 28 - August 2 2019, Association for Computational Linguistics.
[19] Yang, Y., Yih, W. and Meek, C., 2015, "WikiQA: A Challenge Dataset for Open-domain Question Answering.” In Proceeding of The 2015 Conference on Empirical Methods in Natural Language Processing”, pages 2013 - 2018, Lisbon, Portugal. Association for Computational Linguistics.
[20] Yu, L., Hermann, K. M., Blunsom, P. and Pulman, S., 2014, "Deep Learning for Answer Sentence Selection”. In NIPS Deep Learning and Representation Learning Work shop, Montreal.
[21] Yunianta, A., Barukah, O.M., Yusof, N., Dengen, N., Haviluddin, H. and Othman, S., "Semantic data mapping technology to solve semantic data problem on heterogeneity aspect,” Int. J. Adv. In- tell. Informatics, vol. 3, no. 3, pp. 161 - 172, Dec. 2017, doi: https://doi.org/10.5120/11638 - 7118.
[22] https://miro.medium.com/max/2800/0*0efgznFIlATZ2qkY , "The Architecture of CNN".
[23] https://www.researchgate.net/publication/337578683_A_Survey_of_Vehicle_Review_Identification_Based_on_Deep_Learning/figures?lo=1, "Schematic diagram of the Siamese Network structure".

Fig. 13. The Chart of Epoch versus F1-Score for Triplet and Contrastive Loss Function