INSTITUTE OF ENGINEERING & MANAGEMENT, KOLKATA

Survey Report on Paraphrase Detection

(Submitted on: 13.04.2023)

Submitted by:

Sujan Ghosh

Stream: CSE

Section: A

Roll no.: 60

Year: 2nd

Enrolment no.: 12021002002123

Sangeet Bose

Stream: CSE

Section: A

Roll no.: 64

Year: 2nd

Enrolment no.: 12021002002127

Abstract

This paper focuses on paraphrase detection, which is widely studied in the field of natural language detection in NLP. With the development of neural models, experimentation with paraphrase detection in recent years has resulted in a gradual transition to neural methods. This provided models for the perfect presentation of input content and the detection of easy, diverse, and human paraphrases. This article deals with different detection paraphrasing procedures with the main goal of neural methods.

Key-Words: Datasets, Evaluative methods, Traditional Methods, Neural Approaches, Comparative Studies

Introduction

To paraphrase is to "put something written or spoken in different words, especially in a shorter and simpler form, so as to make the meaning clearer" (Cambridge Online Dictionary, 2022). NLP is an industry that makes a computer understand, process and create human language. It is widely used in chatbots and plagiarism detectors. Paraphrase detection is one of the basic operations in the field of NLP. Paraphrase detection checks whether two statements have the same meaning or not. NLP and machine learning techniques help in paraphrase detection. For example, the question How far is the Earth from the Sun can be phrased in the same way as What is the distance between the Earth and the Sun To the best of our knowledge, this is our first survey on neural methods for paraphrase detection. Therefore, our goal in this paper is to provide an overview of paraphrase detection with the main goal of neural methods. In the following section, we will first introduce the most common and widespread datasets for paraphrase detection (Part 2). We then point out the traditional evaluation metrics and show some of the traditional approaches that are used before neural methods. Neural models, the focus of this paper, will be shown. After showing all the methods, we will compare the performance of different models for paraphrase detection.

SENTENCES	PARAPHRASES
How do I improve my English	What is the best way to learn the English
How far is Earth from Sun	What is the distance between Earth and Sun

Table 1: Examples of paraphrases from available datasets for paraphrase detection. (Jianing Zhou and Suma Bhat)

There are several ways we can contribute to this paraphrase detection research. Some of the ways are:

- We can develop a new and unique machine learning model or an algorithm for paraphrase detection. We can explore new procedures to identify paraphrasing and redefine existing machine learning models for better accuracy and performance.
- We can collect and can also modify a high-quality dataset for paraphrase detection. We can create a dataset that captures various directions of paraphrasing which are syntactically and semantically similar. We can use it to access the performance of existing models and new models that we can develop.
- We have to conduct studies to understand the characteristics of paraphrasing. We can access the features that distinguish paraphrases from non-paraphrases, such as lexical and syntactic patterns, and use the insights to improve paraphrase detection models.
- We need to evolve new evaluation metrics to measure the performance of paraphrase detection models. We can also create metrics that are more robust and informative than existing metrics. The existing metrics often rely on human judgments.
- We can study the applications of paraphrase detection in detail in various fields like NLP, information retrieval, and machine translation.

Challenges with paraphrase detection on user generated short texts, such as Twitter, include language irregularity and noise. To cope with these challenges, we propose a novel deep neural network-based approach that relies on coarse-grained sentence modelling using a convolutional neural network and a long short-term memory model, combined with a specific fine-grained word-level similarity matching model. Our experimental results show that the proposed approach outperforms existing state-of-the-art approaches on user-generated noisy social media data, such as Twitter texts, and achieves highly competitive performance on a cleaner corpus.

Datasets

There are several datasets used in paraphrase detection which shall discussed in this part.

PPDB The paraphrase database (Ganitkevitchet al., 2013) contains more than 220 million paraphrase pairs. It also consists of 73 million phrasal and 8 million lexical phrases. It also contains 140 million paraphrase patterns. These patterns always result in meaning-preserving and syntactically correct transformations. Each pair of this data set has a set of associated scores that also includes paraphrase probabilities and also contains a monolingual distribution. The use of this dataset has decreased significantly in recent years because it contains paraphrases without sentence paraphrase.

MSCOCO MSCOCO (Lin et al., 2014) is a large dataset for object detection. Human Annotated Captions contains more than 120,000 images, and each image has five captions from five different annotators. About 500 thousand paraphrase pairs are present in this dataset. Annotators generally provide us with the most distinct and clear object as well as the image object.

WikiAnswer This dataset (Fader et al., 2013) consists of about 18 million word-aligned pairs of questions that are often used as paraphrases. The word alignment given by this data set can give words that have the same meaning in the paraphrased sentences. One disadvantage of this data set is that it limits paraphrases to questions only.

Twitter URL Twitter URL (Lan et al., 2017) is created by collecting extensive paraphrases from Twitter. It links tweets through shared URLs. Two subsets are generally present in this dataset, which contain both paraphrases and non-paraphrases. Human annotators annotate one subset and the other is automatically annotated. Annotation is not good because they are tagged automatically.

Quora This dataset was released in 2017. More than 400k rows of duplicate question pairs. 150k pairs are used annotated as paraphrases. These valid question pairs are used in training and testing.

Evaluation methods

Automatic Evaluation

Existing automatic evaluation metrics are commonly NMT metrics such as BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), METEOR (Banerjee and Lavie, 2005) and TER (Snover et al., 2006). BLEU is used for machine translation systems. ROUGE helps with text summarization, which is primarily used in paraphrase detection. ROUGE-N computes the n-gram recall and ROUGE-L computes the longest common subsequence. METEOR helps find BLEU's weakness. SelfBLEU calculates the BLEU score between the generated paraphrase and the source sentence. Reference-based metrics can be converted to non-reference metrics by replacing the reference with the source sentence. The hybrid metric evaluates both the source and the reference. Specifically, the iBLEU score deletes the repetition of the source sentence in a particular paraphrase. TER measures the number of edits needed to convert a translation to match the reference translation. TER is given as a percentage, where lower gives better results.

Human Evaluation

This is important because it provides an accurate and quantitative evaluation of the metrics. On the other hand, automatic evaluation mainly focuses on n-gram overlays. Human evaluation benefits us both accurately and with better quality. In human evaluation, human annotators are used to evaluate the generated paraphrases according to various quality dimensions such as similarity, clarity, fluency. Human evaluation is more expensive than automated evaluation. Human evaluation has greater quality in generating the desired output.

Traditional Methods

In this part we shall discuss about the traditional methods for a paraphrase detection.

Rule-based paraphrase

Rule-based paraphrase detection deals with automatically collected paraphrase rules, which are generally created by hand. Some researchers today want to have paraphrase rules automatically. The limitation of extraction methods results in the generation of long and complex paraphrase patterns, which has a huge impact on performance.

Thesaurus-Based Approaches

This approach usually creates paraphrases by replacing some words in the source sentences with their synonyms extracted from a thesaurus (Bolshakov and Gelbukh, 2004; Kauchak and Barzilay, 2006). Thesaurus-based approaches are used by first extracting from the thesaurus all synonyms for the words to be replaced. This data set is not that diverse. Its use is therefore limited nowadays.

SMT-Based Approach

This approach is primarily concerned with statistical machine translation (SMT). Paraphrase detection can be considered a special case of machine translation. Basically, the argmax function helps to find the argument that gives the maximum predicted value from the target class that has the highest predicted probability. This can be used to create any model of machine translation as well as paraphrase detection. A machine translation model finds the best translation of a text in a certain language f into a text in a language e using statistical translation model p(fje) and a language model p(e): (Jianing Zhou and Suma Bhat)

$$\hat{e} = arg\max_{e \in e^*} p(f|e)p(e)$$

Applying this idea to paraphrase detection, such a model will give us the best paraphrase ^t of a text in the source side s to a text in the target side t
obtained as,

$$\hat{t} = arg\max_{t \in t^*} p(s|t)p(t)$$

5 Neural Approaches

In the past, paraphrasing mainly focused on template-based or statistical machine translation approaches. Template matching a
statistical translation model modelling are very challenging tasks. Many researchers have begun the widespread use of neural models for
paraphrase detection. We will now discuss the neural approaches essentially required for paraphrase detection.

- **Encoder-Decoder Architecture**

Most of the existing paraphrase generation models are normally based on sequence-by-sequence models that consist of an encoder and a decoder. The encoder encodes the source texts into a contextualized vector representation, which contains a list of vector representations that capture the semantics of each word. Then the decoder detects the paraphrases based on the vectors given by the encoder.

Encoding Side

The main function of encoding is to extract semantic information for the decoder to detect

paraphrases. Researchers also have several options for the encoder after several neural models have been developed. Encoder With the consistent goal of learning a better abstract contextual representation of the input text, many architectures have been explored by researchers. (Prakash et al., 2016) first utilized the seq2seq model, which can be implemented as recurrent neural networks, which are generally long-short-term memory (LSTM) networks. A convolutional neural network (CNN) was also implemented to design seq2seq models. CNN has some parameters. It is faster to train (Vizcarra and Ochoa-Luna, 2020). The Transformer model (Vaswani et al., 2017) demonstrated state-of-the-art performance on multiple text detection tasks. Transformer, with its added ability, gives us long dependencies in sentences. In recent days, large language models are used transformer architectures have led to the current state artistic results for several NLP tasks. Researchers now use less controlled data than before.

Decoding Side

A contextualized representation of paraphrases is used in each decoding step on the decoding side with a vector representation of the previously generated words. Finally, a vocabulary distribution is obtained and the word with the highest probability is generated. This method is greedy decoding (Jianing Zhou and Suma Bhat). In addition, a more commonly used method called ray tracing is also currently used (Wiseman and Rush, 2016). Identifies the k-best paths up to the current time step during decoding. Greedy decoding and beam search methods are general approaches for all text detection tasks. Several blocking mechanisms were derived to prevent the decoder from detecting the same words in the source sentences, thereby avoiding the words in the source sentences.

- **Improvements Based on Encoder and decoder architecture**

Several methods that have been used to improve the Encoder-Decoder architecture for paraphrase detection can be clearly divided into two types based on their target: 1) Model Focused and 2) Attribute Focused. We will then imagine them more clearly.

Focused on the model

Model-cantered model improvement only focuses on using various mechanisms to improve the encoder or decoder without paying due attention to the attributes of the detected paraphrases.

Attention

The attention mechanism (Bahdanau et al., 2015) helps the decoder to focus on some phrases or words with high relevance in word detection. First, the importance of each feature in the source sequence at each time step is determined. It emphasizes important information from the input and de-emphasizes less important information. This extra input vector, which is known as the context vector, is provided to the decoder to calculate the weight distribution among all characters.

Copy

A network of pointers was designed to combat the effect of very rare words. The pointer network copies an element from the input sequence directly to the output. The copy mechanism copies several elements from the input sequence specified by the attention mechanism directly into the output sequence. With the copy mechanism, the decoder determines whether it detects mode or copy mode should be used in each step. It was first introduced by Gu et al. (2016) for an abstract summary. Despite the advantage of detecting well-formed paraphrases using the copying mechanism, this has the undesirable consequence of giving the paraphrase a lot of the phrases in the original sentence, but reducing the variety.

Variational Auto Encoder (VAE)

VAE (Kingma et al., 2014) can learn nonlinear representations for high-dimensional inputs. The VAE decoder can provide realistic outputs conditional on the latent representation as well as the learned distribution. Learning is achieved through revision original latent code input from (Jianing Zhou and Suma Bhat). So with the help of VAE, the paraphrase patterns are encoded into a latent representation that gives the model control over the capacity of the learned distribution. Several paraphrase patterns and related phrases or words are grouped under the same latent assignment. VAE is primarily used in deep learning to reduce higher dimensional data to a lower dimension.

Reinforcement learning:

As pointed out (Ranzato et al., 2015), a known problem encoder and decoder architecture is exposure bias: the decoding of the current word is conditioned by gold references during training but on generated output from the last time step during testing. Therefore, error can accumulate and spread during testing. Another problem lies in mismatch between training goal and assessment metrics. While the generated paraphrases are in the end, it automatically evaluates using the previous one of the mentioned metrics, the network is trained to maximize the probability of generating a reference paraphrase. Hence the minimization of training loss may not correspond to ranking optimization metric. To solve this limitation, amplification learning (RL) is used. RL aims to train an agent that interacts with the environment to maximize its reward. Towards finding optimal policy, RL can be used to maximize reward marked as a required evaluation metric or combination of multiple desired metrics. Rather than minimizing losses (conventional approach), Li et al. (2018) used RL for the first time to maximize the reward given by the evaluator that yields the actual outcome value representing the degree of agreement between the two sentences as paraphrases of each other. Another reward functions have been investigated by researchers, including ROUGE scores, confusion scores, and language fluency (Siddique et al., 2020; Liu et al., 2020).

Generative adversarial networks (GAN):

Designed by Goodfellow et al. (2014), GANs consist of generators and discriminators, where generators try to generate realistic outputs that correspond to reality distributions and discriminators try to distinguish between samples generated by generators and samples that are real. The GAN is initially trained minimax optimization proposed in (Goodfellow et al., 2014). However, when GAN is applied in text generation, the traditional training method

cannot because generating discrete words is non-differentiable. Hence the idea of politics gradient (Sutton et al., 1999) is used to solve this problem (Yu et al., 2017). With the gradient of politics discriminators function as reward functions in RL. In addition, they can provide different discriminators various required rewards and thereby equip model with the capacity to generate text with different conditions. This is where the model is usually trained in the opposite way: generators and discriminators they are pre-trained first, then the generators are trained maximize the loss of fixed discriminators, then the generators are fixed and the discriminators are retrained to minimize loss by providing real samples and samples generated by marker generators. For the paraphrase generation task, different discriminators are designed to differentiate between generated samples and real samples, paraphrase and non-paraphrase (Yang et al., 2019; Vizcarra and Ochoa-Luna, 2020).

Attribute focused:

For attribute-focused enhancements, their purpose is to improve the quality of generated paraphrases in some particular aspects and provide control over some of the generated attribute's paraphrase, such as the level of syntax and granularity. These attribute-focused works typically use priors listed models as their backbone models. Based on backbone models, different mechanisms are used for different orientations. Diversity Attempts to focus on diversity have a goal generate multiple different paraphrases for a given thing sentence. Some works control diversity by providing control signals to the decoder. Random pattern insertions are used by (Xu et al., 2018). (Kumar et al., 2019) used a submodular mechanism maximize submodular fidelity measurement functions and diversity. (An and Liu, 2019), (Chen et al., 2020) and (Cao and Wan, 2020) all generate various paraphrases by providing a decoder with different latent patterns as the control signal. Further (Cao and Wan, 2020). their diversity-loss model for diversity control. (Liu et al., 2020) use RL with multiple reward functions create different paraphrases. One of reward function calculates the ROUGE score between the generated sentence and the original sentence which can focus on word variation and variety. Acting as a reward function in RL, discriminators naturally can be used to ensure control over some required attributes. (Qian et al., 2019) used more generators in GAN generate more diverse paraphrase. A generator discriminator is used for this distinguish sentences generated by different generators and guarantee that the paraphrases produced are sufficiently diverse.

Word-Level Work on word-level paraphrasing focuses primarily on generating paraphrases by replacing original words in source texts with synonyms. Some works used external linguistic knowledge (Cao et al., 2017; Lin et al., 2020). (Cao et al., 2017) used an alignment table capturing many synonym mappings based on the IBM model (Chahuneau et al., 2013). (Lin et al., 2020) used WordNet (Miller, 1995) to find synonyms. Other works instead proposed special mechanisms to learn synonym mappings (Ma et al., 2018; Fu et al., 2019). For example, (Ma et al., 2018) used a search-based method to learn such a mapping. (Fu et al., 2019) incorporates a novel latent bag-of-words mechanism into a seq2seq content planning model that primarily provides candidate synonyms for words in source texts. However, generating a paraphrase only at the word level limits the quality and variety of generated paraphrases. Therefore, paraphrasing was also studied at another level of granularity, e.g. the level of syntax.

Syntax This category explores methods that provide control over the syntax of generated paraphrases. In principle, all the methods used in the previous works can be divided into two classes: 1. Explicit checking and 2. Implicit checking. Methods in the first class first encode the syntactic tree of an exemplar sentence into a list of vector representations and then add them to the decoder at each time step during decoding (Iyyer et al., 2018; Chen et al., 2019; Goyal and Durrett, 2020; Kumar et al.., 2020). These methods can provide explicit control over the syntax of the generated paraphrases and thus have better interpretability. The second class of methods first learns how to distribute syntax information using VAE. Then, the latent syntactic variable sampled from the learned distribution will be fed to the decoder at each decoding step (Chen et al., 2020). Although the control provided by this method is implicit, it does not require pattern sentences and can also group multiple related syntaxes under the same latent assignment.

Observation:

Paraphrase detection is an NLP classification problem. Given a pair of sentences, the system determines the semantic similarity between the two sentences. If both sentences convey the same meaning, then it is marked as a paraphrase; otherwise, it is marked as a non-paraphrase. Most existing paraphrase systems have performed quite well on plain text corpora such as the Microsoft Paraphrase Corpus (MSRP) (Dolan, Quirk, & Brockett, 2004). However, paraphrase detection in noisy user-generated tweets is more challenging due to issues such as typos, acronyms, style, and structure (Xu, Ritter, Callison-Burch, Dolan, & Ji, 2014). Moreover, measuring semantic similarity between two short sentences is very difficult due to the lack of common lexical features (Kajiwara, Bollegala, Yoshida, & Kawarabayashi, 2017). Although paraphrase detection in noisy short texts has received little attention to date, some initial work has been reported on the SemEval 2015 Twitter benchmark dataset (Dey, Shrivastava, Kaushik, 2016, Xu, Callison-Burch, Dolan, 2015, Xu, Ritter, Callison- Burch, Dolan, Ji, 2014). Unfortunately, the best performing approaches on one data set do not seem to perform as well when evaluated against another. As we discuss later in this paper, the state-of-the-art approach for the SemEval dataset proposed by Dey et al. (2016) does not perform well (in the form of F1 scores) when evaluated on the MSRP dataset. Similarly, Ji and Eisenstein (2013) are the best performing approach on the MSRP dataset but does not perform well on the SemEval dataset. In conclusion, existing approaches are not very general, but rather highly dependent on the data used for training.

Comparative Study:

The datasets used in this survey report are PPDB, MSCOCO, WikiAnswer, Twitter URL and Quora. In the dataset PPDB, millions of lexical phrases are present. MSCOCO on the other hand, is a large-scale object dataset. Moreover, it is the most prominent dataset as annotators are used and gives the object of an image. WikiAnswer consists of word-aligned question pairs not like PPDB and MSCOCO. The disadvantage of this dataset is that it is only limited and restricted to the questions only for the purpose of paraphrase detection. Twitter URL also uses annotators just like MSCOCO but the annotation in this case is not so good as the

annotators are labeled automatically. In Quora, 150k pairs are used annotated as paraphrases and the valid question pairs of this dataset are used in training and testing.

Although recent neural models have shown great progress, the state-of-the-art results are still not satisfactory enough. Therefore, there is still a need to explore more advanced paraphrasing models. Below we discuss several potential lines of research that we believe are worth exploring.

Pretrained language models

Virtually all recent work related to the application of pre-trained language models to generate paraphrases is quite naive. Therefore, we could combine large pre-trained language models with other mechanisms such as reinforcement learning, VAE and GAN.

Multilevel controllable paraphrase generation

Recent work on multi-level paraphrase generation focuses only on word-level paraphrasing and phrase-level paraphrasing. However, multiple levels of granularity can be incorporated. We believe that it is worthwhile to study a combination of different levels, including word level, phrase level, syntax level, and sentence level.

Transfer learning

With the goal of generating different surfaces of given sentences while preserving meaning, text summarization, text simplification, and paraphrase generation are essentially similar. Therefore, transfer learning of these three tasks could be used to improve performance. Generating Stylistic Paraphrase Currently, word and phrase substitution cannot be carefully controlled when generating paraphrases. Therefore, it is difficult to control the style of the generated paraphrases. We believe it is worth exploring methods of incorporating specific styles into the generated paraphrases. For example, by controlling word and phrase types, we can incorporate metaphor and idiomatic expressions into paraphrases (Zhou et al., 2021b,a), which could also help increase the creativity and variety of generated paraphrases.

Evaluation metrics

As discussed in earlier, the BLEU score and other automatic evaluation metrics based on a similar principle are not good enough to evaluate paraphrase generation. So there exists a

the need for better automated evaluation methods. One possible method is to use paraphrase identification in automatic evaluation metrics to explicitly evaluate whether the generated sentence and the input sentence are paraphrases.

Conclusion:

We described an approach to incorporate an AMR parser output into the detection of paraphrases. Our method works by merging two graphs that need to be tested for a paraphrase relation, and then re-weighting a sentence-term matrix by the PageRank values of the nodes in the merged graph. We find that our method gives significant improvements over state of the art in paraphrase detection in the transudative setting, showing that AMR is indeed

helpful for this task. We further show that the inductive settings are instead not ideal for this type of approach.

References:

- Fernando, S., & Stevenson, M. (2008, March). A semantic similarity approach to paraphrase detection. In *Proceedings of the 11th annual research colloquium of the UK special interest group for computational linguistics* (pp. 45-52).
- Socher, R., Huang, E., Pennin, J., Manning, C. D., & Ng, A. (2011). Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. *Advances in neural information processing systems*, *24*.
- Agarwal, B., Ramampiaro, H., Langseth, H., & Ruocco, M. (2018). A deep network model for paraphrase detection in short text messages. *Information Processing & Management*, *54*(6), 922-937.
- Issa, F., Damonte, M., Cohen, S. B., Yan, X., & Chang, Y. (2018, June). Abstract meaning representation for paraphrase detection. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (pp. 442-452).
- Cordeiro, J., Dias, G., & Brazdil, P. (2007, March). A metric for paraphrase detection. In *2007 International Multi-Conference on Computing in the Global Information Technology (ICCGI'07)* (pp. 7-7). IEEE.