Vol.2020-NL-246 No.29
Vol.2020-SLP-134 No.29
2020/12/3

Cost-Friendly Feature-based Approach for Paraphrase Identification

Xiaodong Liu[1,a)] Rafal Rzepka[2,b)] Kenji Araki[2,c)]

Abstract: Among approaches utilizing features to address paraphrase identification, ELMo-based methods and task-specific DNNs have achieved competitive performance. However, their consumption of resources for pre-training is still considerable. Our approach, on the other hand, consumes substantially less resources while preserving the similar performance level. To implement our method, we utilize features representing multiple levels of granularity: semantic similarity at word, phrase and sentence levels. While being light on resources, our method achieves fairly competitive results on Microsoft Research Paraphrase Corpus (MRPC) compared to task-specific DNNs. To confirm our method is not over-fitting to the MRPC, we propose a novel robust test on Quora Question Pairs (QQP). In addition, to explore the capability of our method beyond binary classification, we apply it to a textual entailment task (SICK-E). Similar accuracy to a recently published ELMo-based method is achieved.

Keywords: paraphrase identification, multiple levels of granularity, semantic similarity, cost-friendly, cost-effective

1. Introduction

Measuring semantic similarity is crucial to a certain number of language processing tasks such as plagiarism detection, query ranking, and question answering. In this paper, we propose a cost-friendly feature-based approach to modelling semantic similarity between sentences, which can determine whether a pair of sentences is a paraphrase.

Among approaches utilizing features to address Paraphrase Identification (PI), ELMo-based methods and task-specific DNNs have achieved competitive performance. However, with the advent of BERT [Devlin et al., 2019] and its successors like XLNET [Yang et al., 2019], fine-tuning on those pre-trained models has achieved state-of-the-art results on many downstream tasks, including PI. ELMo-based methods and task-specific DNNs, while also having extensive parameter complexity and requiring long training process on large corpora, have become less cost-effective. Especially when it comes to some task-specific DNNs like CNNs and RNNs, the pre-trained models tailored only for a specific task double-depreciate their cost-effectiveness.

Moreover, a recent study [Shi et al., 2019] on ELMo [Peters et al., 2018] shows that in many cases, the contextualized embedding of a word changes drastically when the context is paraphrased. As a result, the downstream model is not robust to paraphrasing and other linguistic variations. To address the problem, the study employed a paraphrase-aware retrofitting (PAR) method with external paraphrase corpora to enhance the stability of contextualized models on sentence similarity tasks. The performance of the retrofitted ELMo increases commensurately with the size of corpora. Nonetheless, in addition to extensive parameter complexity and long training process on large corpora, existing paraphrase corpora with human annotation are also limited resources.

Our approach, compared to ELMo-based methods and task-specific DNNs, consumes substantially less resources while preserving the similar performance level. With limited consumption of resources, although our approach has not outperformed fine-tuning on those transformer-based models, it can bring back feature-based approaches for PI to the level of cost-effectiveness.

There is a feature-based approach for PI that has achieved better performance than ours (see in Section 6.1), and also consumed less resources (pre-training is not required): utilizing TF-KLD weighting scheme proposed by [Ji and Eisenstein, 2013] to generate discriminative semantics for sentence representations. Nevertheless, their method is strictly confined to one dataset (MRPC [Dolan et al., 2004]), and can not go beyond binary classification. Therefore, in this work, we test our method not only on the MRPC, but also on the QQP [Iyer et al., 2017] for a novel robust test. Furthermore, to explore if our method can go beyond binary classification, we apply it to a textual entailment task: SICK-E [Marelli et al., 2014].

To implement our method, we adopt the same strategy to model sentence similarity as some previous works [Socher et al., 2011][Yin and Schütze, 2015a][He et al., 2015] did – to compare two sentences on multiple levels of granularity. This work addresses granularity in two parts: a) core features for semantic similarity at sentence level; b) engineered features for other considerations including semantic similarity at word and phrase levels. The rationale of utilization of these features is described

[1] Graduate School of Information Science and Technology, Hokkaido University, Sapporo, Japan
[2] Faculty of Information Science and Technology, Hokkaido University, Sapporo, Japan
[a)] xiaodongliu@ist.hokudai.ac.jp
[b)] rzepka@ist.hokudai.ac.jp
[c)] araki@ist.hokudai.ac.jp

Approaches	Merits	Demerits
(1) Feature Engineering	semantic similarity at word and phrase levels	semantic similarity at sentence level
(2) Distributional Models	latent representation with low dimensionality & aggregated observable variables	lack of synonym recognition or lexical knowledge on a small corpus
(3) Weighting Scheme	discriminative sentence semantics	it could be confined to corpus size & binary classification
(4) Task-specific DNNs w/ pre-training	automatically extracted features from different levels of granularity	time-costly pre-training for a specific task; automatically extracted features might not be compatible with (1)
(5) Task-specific DNNs w/o pre-training	same as (4) but with friendly time cost for downstream tasks	back-propagation run for a new task; model could be over-fitting to a task; same compatibility issue as (4)
(6) ELMo-based Methods	discriminative word semantics to address homonyms	requirement of additional task-specific architectures; drastic change of word semantics when the context is paraphrased [Shi et al., 2019]

Table 1: Merits & demerits of various feature-based approaches for paraphrase identification.

in Section 2. The core features are derived from sentence latent representations that are generated by our two pre-trained latent spaces. The detailed implementation of pre-training latent spaces is described in Section 3. To show the strength of our pre-trained latent spaces for sentence latent representations, Section 4 is presented. In Section 5, we list the core features and engineered features for the experiments in Section 6.

2. Motivation and Related Work

A number of feature-based approaches for PI, from feature engineering to task-specific DNNs, implicitly or explicitly have stressed the importance that when it comes to PI, it is essential to compare two sentences on multiple levels of granularity. [Wan et al., 2006] proposed some fine-grained features, in which n-gram overlap for semantic similarity of words and phrases and tree edit distance for semantic similarity of sentences. The TF-KLD proposed by [Ji and Eisenstein, 2013] utilizes weighting scheme for discriminative sentence semantics, plus the n-gram overlap [Wan et al., 2006] for semantic similarity at word and phrase levels. The recursive autoencoder [Socher et al., 2011] and the BI-CNNs [Yin and Schütze, 2015a] [He et al., 2015] are used to automatically extract features from different levels of granularity. Their merits and demerits are listed in Table 1.

As shown in Table 1, (1), (2), and (3) are the most cost-friendly feature-based approaches, which do not require pre-training and can be implemented on a single CPU device. We take their merits into consideration to address granularity for the realization of our approach: The demerit of (1) can be offset by the combination of (2) and (3) merits. Unlike the TF-KLD [Ji and Eisenstein, 2013] weighting scheme that is strictly confined to the MRPC and binary classification, the demerit of (3) does not fit in our dual weighting scheme, which is proved in Section 6.1. TF-KLD-KNN proposed by [Yin and Schütze, 2015b] can address the limitation of weighting unseen words, but it still cannot go beyond binary classification. As for the demerit of (2), we shift the burden of lexical knowledge to word-embeddings. We use fastText [Mikolov et al., 2018] trained with subword information on CommonCrawl for its better performance compared to GloVe [Pennington et al., 2014] trained on CommonCrawl in SentEval framework [Conneau and Kiela, 2018].

3. Latent Space Pre-training

In this section, we describe how to pre-train the proposed latent spaces. The pre-trained spaces then are further used for the generation of sentence latent representations, which are the core features for measuring semantic similarity at sentence level. The diagram of pre-training architecture is shown in Figure 1, which describes the mapping relationship between weighted components and latent representations. Details about latent representation, components, dual weighting scheme, and two-layer feedforward neural networks are described in the corresponding subsections.

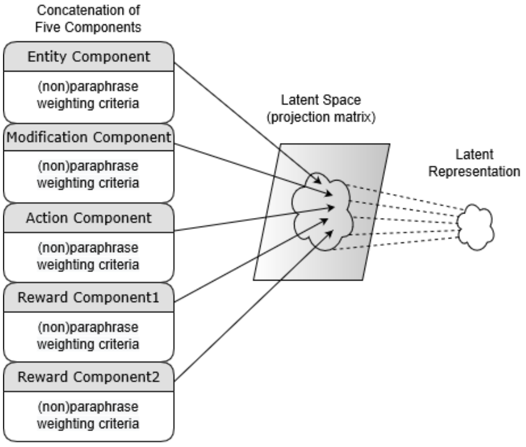
Concatenation of Five Components
Entity Component
(non)paraphrase weighting criteria
Modification Component
(non)paraphrase weighting criteria
Action Component
(non)paraphrase weighting criteria
Reward Component1
(non)paraphrase weighting criteria
Reward Component2
(non)paraphrase weighting criteria
Latent Space (projection matrix)
Latent Representation

Fig. 1: The mapping diagram of our model architecture.

The corpus that we use for pre-training and distributional models is only the training dataset of the MRPC [Dolan et al., 2004] consisting of 4,076 sentence pairs, in which 2,753 pairs are labeled as paraphrase. With the small training corpus plus the simple model architecture (two-layer feedforward neural network), the pre-training process can be completed within 3 hours using a GPU device (NVIDIA RTX 2080 TI) or a single day using a CPU device (Intel i7-6700k).

3.1 Latent Representation

We use distributional models with matrix factorization to generate latent representations for all sentences of the training dataset in the MRPC. For factorization, we choose Singular Value Decomposition [Deerwester et al., 1990] provided by Scikit-learn tool, and latent dimensionality is set as 100 [Guo and Diab, 2012]. Moreover, we normalize the factorized matrix, as the activation function used in our feedforward NN is tanh. As a result, all values in the factorized matrix are included within an open interval (-1, 1).

Furthermore, we refine the latent representations for paraphrase pairs. Suppose the latent representations for a sentence S_1 and a sentence S_2 of a sentence pair are (0.9, 0.8, 0) and (0.7, 0.8, 0.6) respectively, the pair share a latent representation by averaging the elementwise addition of two vectors, in this case, the shared latent representation is (0.8, 0.8, 0.3). The rationale of sharing is that paraphrase pairs should have similar or even identical dimension values. By doing so, significantly uneven values between particular dimensions of a sentence pair can be adjusted; e.g. in this hypothetical case, the value for the third dimension of the first vector is 0 originally, and 0.3 is assigned after the operation, which is the middle-ground for both 0 and 0.6.

3.2 Five Components

There are three word-embedding components and two reward components: Entity, Modification, Action, Reward Component 1, and Reward Component 2. For word-embedding components, they are created based on three specific sets of part-of-speech tags (POS-tags) provided by the NLTK.

Entity set: singular noun, plural noun, singular proper noun, plural proper noun, personal noun

Modification set: determiner, predeterminer, adjective, comparative adjective, superlative adjective, possessive pronoun, possessive ending, existential there, modal, adverb, comparative adverb, superlative adverb, particle, to

Action set: base form verb, past tense verb, present particle verb, past participle verb, present verb, third person present verb

Suppose the sentence “He likes apples.” has three word embeddings (0.1, 0.1, 0.1) for ‘he’, (0.2, 0.2, 0.2) for ‘likes’, and (0.3, 0.3, 0.3) for ‘apples’, then **Entity Component** is [(0.1, 0.1, 0.1) + (0.3, 0.3, 0.3)] / 2 = (0.2, 0.2, 0.2); **Action Component** is (0.2, 0.2, 0.2) / 1 = (0.2, 0.2, 0.2); **Modification Component** is (0, 0, 0). If the word embedding of a word does not exist in the embedding space, our strategy is to skip it, because although some sentences of the MRPC are relatively long with noise, they are derived from old news resources rarely containing important semantics of newly-coined words like ‘infodemic’.

Each **Reward Component** is initiated as a vector with the same length (300) as fastText word embeddings, and the values of all elements are assigned 1.0. Two weighted Reward Components are meant to provide strong or weak rewards in the concatenation of components. With their impact, sentence pairs that have paraphrase/non-paraphrase-like characteristics tend to obtain similar latent representations in paraphrase/non-paraphrase spaces.

3.3 Dual Weighting Scheme

We first introduce the weighting criteria in Section 3.3.1. In Section 3.3.2, we describe how to decide and tune weights.

3.3.1 Weighting Criteria

The MRPC [Dolan et al., 2004] is a widely-adopted paraphrase corpus for testing various methods, which is also incorporated in the GLUE benchmark [Wang et al., 2018]. Thus, we utilize the characteristics of its training dataset to determine some of our weighting criteria.

The *sent-len* (Figure 2a) and *abs-sent-len-diff* (Figure 2b) are comparatively less discriminative, so we combine them together to weight Reward Component 1.

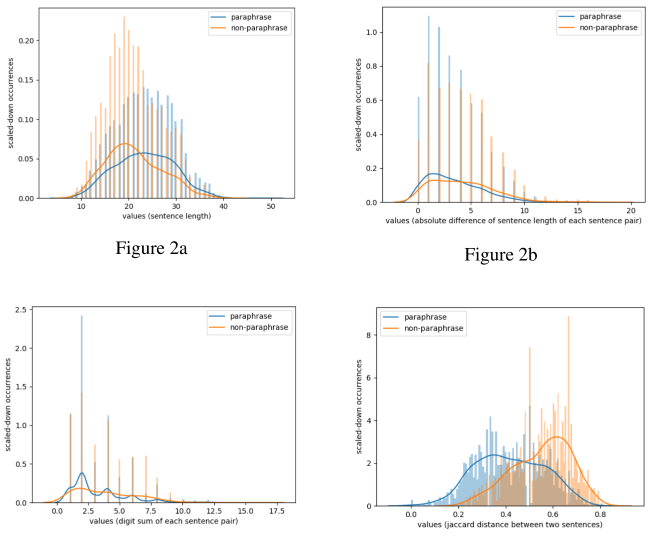

Figure 2a

Figure 2b

Figure 2c

Figure 2d

Fig. 3: Based on four MRPC characteristics, thresholds of occurrence differences are: 23 for sentence length (*sent-len*) (Figure 2a); 5 for absolute difference of sentence length in tokens (*abs-sent-len-diff*) (Figure 2b); 4 for digit-sum in both sentences (Figure 2c); and 0.6 for Jaccard distance (Figure 2d).

As for Jaccard distance [Jaccard, 1912] in Figure 2d, we carried out an experiment to identify paraphrase of the MRPC using the distance, and found out at 0.6, accuracy and F-score increase significantly. Thus, it is considered as a criterion to weight Reward Component 2.

For three word-embeddding components, as adverbs like negation (not) present discriminative semantics, we assign more weight to Modification Component if a sentence pair is labeled as non-paraphrase. Semantic similarity of nouns and verbs is the basic requirement for semantic similarity of sentences, so we assign more weight to Entity and Action Components if a sentence pair is labeled as paraphrase.

The digit-sum is a criterion to affect the weight of Entity Component (only sums exceeding 0 are illustrated in Figure 2c). We assume when there are too many digits occurring in a sentence pair, they would blur the semantics of Entity Component. To confirm if this assumption can affect all tasks in Section 6, for comparison, another two latent spaces are also pre-trained with dual weighting scheme **not** containing digit-sum. For readability, two types of pre-trained latent spaces are abbreviated as the *‘with digit-sum’* and the *‘without digit-sum’* respectively in the rest of this paper.

3.3.2 Weights Tuning

The dual weighting scheme is summarized in Table 2. There are three types of weights: pair (0.5 & 1.5), strong reward (1), and weak reward (0.2). To decide them, first we hypothesize that 1 is the best fit for strong reward, because if too big it will make the dimensions of other input components less significant; when too small it cannot be considered as a strong reward. Then we keep the pair as it is by default (0.5 & 1.5 in this case), and tune weak reward from 0.1 to 0.3 for best generalization of back propagation. After 0.2 is decided, we start to tune pairs at the pivot of

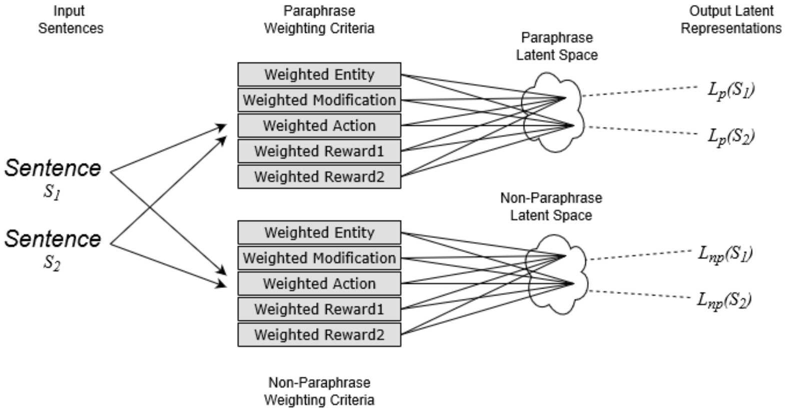

Fig. 4: Sentence latent representations projected from two spaces.

	Paraphrase Weighting	**Non-paraphrase Weighting**

Entity	if digit-sum ≤ 4:
	Entity = 1.5 * Entity Component
	else:
	Entity = 0.5 * Entity Component
		if digit-sum > 4:
		Entity = 1.5 * Entity Component
		else:
		Entity = 0.5 * Entity Component

Modi.	*Modi.* = 0.5 * Modification Component	*Modi.* = 1.5 * Modification Component
Action	*Action* = 1.5 * Action Component	*Action* = 0.5 * Action Component

RC_1	if *sent-len* ≥ 23 or *abs-sent-len-diff* ≤ 5:
	RC_1 = 1 * Reward Component 1
	else:
	RC_1 = 0.2 * Reward Component 1
		if *sent-len* < 23 or *abs-sent-len-diff* > 5:
		RC_1 = 1 * Reward Component 1
		else:
		RC_1 = 0.2 * Reward Component 1

RC_2	if Jaccard-distance ≤ 0.6:
	RC_2 = 1 * Reward Component 2
	else:
	RC_2 = 0.2 * Reward Component 2
		if Jaccard-distance > 0.6:
		RC_2 = 1 * Reward Component 2
		else:
		RC_2 = 0.2 * Reward Component 2

Table 2: Dual weighting scheme: for *Entity* of the *'without digit-sum'*, *Entity* = (1.5/0.5) * Entity Component for paraphrase/non-paraphrase weighting.

1.0 from (0.1 & 1.9) to (0.9 & 1.1). (0.5 & 1.5) is the pair that can generate best discriminative similarity (see in Section 4).

3.4 Two-Layer Feedforward Neural Network

For each sentence, the concatenation of five weighted components is mapped to its latent representation. All paraphrase sentences of the training dataset in the MSRP are used to train the paraphrase space based on paraphrase weighting criteria, while the non-paraphrase sentences are used to train the non-paraphrase space based on non-paraphrase weighting criteria. Since the size of training dataset is small, back-propagation is performed for each mapping, following Stochastic Gradient Descent [Bottou, 1998]. The hyperparameters of our feedforward NN are as follows: activation function is tanh; input vector length is 1,500; target vector length is 100; loss function is LSE; epochs is 500; learning rate is 5e-4; gradient descent is SGD.

4. Discriminative Similarity

By the two pre-trained latent spaces, as shown in Figure 4, each sentence of any pair can obtain two latent representations after projection from two spaces: *Lp*(S_1) and *Lnp*(S_1), or *Lp*(S_2) and *Lnp*(S_2). Using all sentence pairs in the MRPC training dataset, we calculate cosine similarity between *Lp*(S_1) & *Lp*(S_2) and cosine similarity between *Lnp*(S_1) & *Lnp*(S_2).

1) With respect to the *'with digit-sum'*, the mean and standard deviation are [0.81, 0.15] and [0.81, 0.14] (paraphrase pairs); [0.68, 0.19] and [0.73, 0.16] (non-paraphrase pairs).

2) With respect to the *'without digit-sum'*, the mean and standard deviation are [0.82, 0.14] and [0.81, 0.14] (paraphrase pairs); [0.71, 0.17] and [0.73, 0.16] (non-paraphrase pairs).

In both cases, similarities of non-paraphrase pairs tend to increase in non-paraphrase space. In contrast, similarities of paraphrase pairs tend to remain the same or decrease.

5. Features

The features designed in this work follow two principles: granularity-based and flexibility-based. Granularity-based features must represent semantic similarity at word, phrase, or sentence level. The independent existence of flexibility-based features can be insignificant to PI, but once they are added upon to corresponding granularity-based features, overall performance should be significantly augmented.

Semantic Similarity at Sentence Level (SSSL)
1 *Lp*(S_1) + *Lp*(S_2)
2 |*Lp*(S_1) - *Lp*(S_2)|
3 *Lnp*(S_1) + *Lnp*(S_2)
4 |*Lnp*(S_1) - *Lnp*(S_2)|
5 cosine similarity between *Lp*(S_1) & *Lp*(S_2)
6 cosine similarity between *Lnp*(S_1) & *Lnp*(S_2)
7 euclidean distance from *Lp*(S_1) to *Lnp*(S_1)
8 euclidean distance from *Lp*(S_2) to *Lnp*(S_2)
Flexibility-based Features for SSSL (FSSSL)
9 |*entity_count*(S_1) - entity_count(S_2)|
10 |*action_count*(S_1) - action_count(S_2)|
11 |*digit_count*(S_1) - digit_count(S_2)|
12 Levenshtein_edit_distance(S_1, S_2)
13 word_mover_distance(entity_action_tokens_S_1, entity_action_tokens_S_2)
Semantic Similarity at Word and Phrase Levels (SSWPL) [Wan et al., 2006]
14 unigram recall/precision
15 bigram recall/precision
16 trigram recall/precision
17 BLEU recall/precision
18 absolute difference of sentence length
Flexibility-based Features for SSWPL (FSSWPL) [Popović, 2015]
19 chrf recall/precision (1-6)

Table 3: Granularity and flexibility-based features.

All features are summarized in Table 3. **SSSL** is our core features inspired by the work of [Ji and Eisenstein, 2013]: 5, 6, 7, and 8 appended to the concatenation of 1, 2, 3, and 4. In **FSSSL**, the edit distance is taken for the consideration of n-gram overlap at sentence level, and the rest is considered to enhance the similarities and distances of SSSL. For **SSWPL**, we utilize some fine-grained features [Wan et al., 2006] to implement it as [Ji and Eisenstein, 2013] did in their work. **FSSWPL** is implemented by the chrf (character n-gram F-score) MT metric [Popović, 2015]. As recommended by the author, the chrf recall/precision is based on weight distributions up to 6-gram: (1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6).

6. Experiments and Results

Three experimental results are described in three corresponding subsections. Here we underline that all the following experiments use two pre-trained latent spaces with dual weighting scheme to generate sentence latent representations (for both training and test dataset), without task-specific re-pretraining or reweighting design. For classification, we choose Support Vector Machine with a linear kernel provided by Scikit-learn tool; parameter C is tuned for best accuracy.

6.1 Paraphrase Identification Task (MRPC)

With respect to the MRPC [Dolan et al., 2004], the distribution of sentence pairs of paraphrase/non-paraphrase in training and test data is (2,753/1,323) and (1,147/578). As shown in Table 4, we test the features using step-by-step addition to observe the effectiveness of our feature design.

Device Requirement	This Work	Acc.(%)	F(%)
CPU/GPU	SSSL	73.6 / 72.2	82.4 / 81.8
	SSSL + FSSSL	75.7 / 75.0	83.3 / 82.8
	SSSL + FSSSL + SSWPL	76.2 / 75.7	83.5 / 83.1
	SSSL + FSSSL + SSWPL + FSSWPL	**77.6 / 78.0**	**84.4 / 84.6**
CPU	**Unsupervised Method**	**Acc.(%)**	**F(%)**
	[Fernando and Stevenson, 2008]	74.1	82.4
	Feature Engineering	**Acc.(%)**	**F(%)**
	[Wan et al., 2006]	75.6	83.0
	[Madnani et al., 2012]	77.4	84.1
	Distributional Models	**Acc.(%)**	**F(%)**
	[Guo and Diab, 2012]	71.5	-
	[Ji and Eisenstein, 2013]	78.6	84.6
GPU(s)	**Task-Specific DNNs**	**Acc.(%)**	**F(%)**
	[Socher et al., 2011] RNN * (baseline)	76.8	83.6
	[Cheng and Kartsaklis, 2015] RNN *	77.5	84.6
	[Yin and Schütze, 2015a] CNN	78.1	84.4
	[Yin and Schütze, 2015a] CNN *	78.4	84.6
	[He et al., 2015] CNN	78.6	84.7
	[Cheng and Kartsaklis, 2015] RecNN *	78.6	85.3
	Hybrid Method	**Acc.(%)**	**F(%)**
	[Yin and Schütze, 2015b] *	78.7	84.8
GPU(s)	**ELMo-based Methods**	**Acc.(%)**	**F(%)**
	[Shi et al., 2019] ELMo-PAR	74.9	-
	[Wang et al., 2018] BiLSTM+ELMo+Attention	78.0	84.4

Table 4: In the table, '-' and '*' denote 'not reported' and 'with engineered features' respectively. The results of this work on the **left/right** side are derived from ***'with digit-sum'*/*'without digit-sum'***.

Based on whole feature sets, best performances of (77.6/84.4) and (78.0/84.6) are achieved respectively for the *'with digit-sum'* / the *'without digit-sum'*. The *'with digit-sum'* outperforms the *'without digit-sum'* in most cases except for the whole feature sets. We believe that the degree of discriminative similarity described in Section 4 causes this difference. Flexibility-based features work for the *'without digit-sum'* significantly in terms of accuracy; |75.0 − 72.2| and |78.0 − 75.7| are substantial. To conclude, the *'without digit-sum'* is comparatively weak on core features representing semantic similarity of sentences, but more compatible with engineered features. The *'with digit-sum'*, on the other hand, shows opposite properties.

Due to the page limit, we list only representative results of previous works, most of which are available at aclweb[*1]. [Socher et al., 2011] pre-trained their recursive autoencoder and utilized engineered features, but their results (baseline in this work) are lower than other task-specific DNNs, and some methods Madnani et al., 2012Ji and Eisenstein, 2013[Madnani et al., 2012], [Ji and Eisenstein, 2013] that are implementable on a CPU device.

Among the methods implementable on a CPU device, The TF-KLD [Ji and Eisenstein, 2013] weighting scheme[*2] used to reweight distributional models is better than ours, but the TF-KLD is strictly confined to the MSRP and binary classification, which is not extensible to multi-classification tasks of sentence pairs such as the SICK-E. The TF-KLD-KNN proposed by [Yin and Schütze, 2015b] can address the limitation of weighting unseen dataset; nevertheless, it is still limited by binary classification. Their main contribution is expanding the embedding space by linguistic phrases, but with added 15 features [Madnani et al., 2012], their improvement on [Ji and Eisenstein, 2013] is insignificant.

As for task-specific DNNs, pre-training is useful to relieving over-tting, which is a severe problem when building DNNs on small corpora [Hu et al., 2014]. In Table 4, most of them inevitably employed external large corpora like English Gigaword [Graff et al., 2003] to pre-train their models, which is time-costly for a specific NLP downstream task. By utilizing engineered features, some of them achieve slightly better results than ours in terms of accuracy. An exception is [He et al., 2015], in which, no pre-training or engineered features are required for their model. However, their model has to run back propagation from scratch for a new task, which is comparatively less efficient in terms of application level. Another likely limitation is the compatibility issue with engineered features: as shown in [Yin and Schütze, 2015a], their performance is insignificantly improved after addition of 15 features [Madnani et al., 2012].

ELMo-based methods need to use task-specific architectures that include the pre-trained representations as additional features [Devlin et al., 2019]. Compared to fine-tuning on BERT [Devlin et al., 2019] (accuracy/F on the MRPC is 84.8/88.9 for BERT-base), while also being extensive parameter complexity and long pre-training process on large corpora, ELMo-based methods for PI are comparatively less cost-effective. In GLUE benchmark [Wang et al., 2018], the best baseline result on the MRPC, as shown in Table 4, is the ELMo plus attention layer. Their result is similar to ours although F score is marginally lower; however, our approach consumes substantially less resources, which is mentioned in the second paragraph of Section 3.

Hypothetical Sentence Pairs Containing Homonyms	Label	Predicted Label
S_1: How is **life** in prison? S_2: I have **life** insurance.	0	0
S_1: The **bear** in the circus is adorable. S_2: I can hardly **bear** to watch the circus.	0	0
S_1: Yesterday I was walking along the river **bank**. S_2: Yesterday I went to the **bank** to withdraw my money.	0	0

Table 5: Three hypothetical sentence pairs containing homonyms.

Additionally, to observe if our method is subject to homonyms, the three hypothetical sentence pairs containing homonyms in Table 5 are tested based on our tuned classifier for the MRPC. They are predicted as true negative for both the *'with digit-sum'* and *'without digit-sum'*.

[*1] https://aclweb.org/aclwiki/Paraphrase_Identification_(State_of_the_art)

[*2] The result of [Ji and Eisenstein, 2013] is better with the help of transductive learning [Gammerman et al., 1998]. We only address paraphrase identication for the case that the test data are not available for training the model.

Data Distributions					**Total Samples**	**Mean Acc.(%)**	**Std. Acc.(%)**	**Mean F(%)**	**Std. F(%)**
Bias	**Training Dataset**		**Test Dataset**						
	Paraphrase	**Non-paraphrase**	**Paraphrase**	**Non-paraphrase**					
Paraphrase Biased	2,753	1,323	1,147	578	30	79.3 / 79.4	0.8 / 0.9	86.0 / 86.1	0.5 / 0.5
Balanced	2,038	2,038	863*	863*	30	72.5 / 72.3	1.0 / 1.0	74.6 / 74.4	1.0 / 0.9
Non-paraphrase Biased	1,323	2,753	578	1,147	30	72.4 / 72.6	0.9 / 0.8	50.5 / 51.3	2.6 / 2.3
Avereage	2,038	2,038	863	863	30	**74.7 / 74.8**	**0.9 / 0.9**	**70.4 / 70.6**	**1.4 / 1.2**

Table 6: The experimental results of robust test on the QQP. By '*', since the test dataset of the MSRP has 1,725 sentence pairs, we reset it to 1,726 for balanced distribution. Total samples are set as 30 because it is the smallest number to estimate normal distribution. The results of this work on the **left/right** side are derived from ***'with digit-sum'/'without digit-sum'***.

6.2 Robust Test (QQP)

Some of our weighting criteria are determined by the characteristics of the MRPC, and two latent spaces are pre-trained on the training dataset of the MRPC. Hence, to confirm our method is not over-fitting to the MRPC, we propose this robust test.

The QQP [Iyer et al., 2017] has over 404k question pairs collected from Quora online resources, in which if a pair of questions is labeled as duplicate, it is a paraphrase pair, and non-paraphrase if not. The original dataset is not divided into train/dev/test subset. Some works divide the dataset using their own distributions for testing their models; e.g. [Tan et al., 2018].

The QQP is selected for this robust test for two reasons: (1) it is substantially larger than the MRPC, and thus random-pick is viable for the three bias distributions in Table 6 (in this work, for each sample, a sentence pair with same id is not allowed to occur more than once; for random-pick, we use Python random package to select ids of pairs); (2) the corpus contains more lexical knowledge than the MRPC does (in this work, we do not filter out any noisy texts, which is conducive to robust test). The size of each sample is the same as the MRPC (5,801), except in the category of 'Balanced', we reset it to 5,802 for even distribution.

Same way for the MRPC, our classifier is tuned for best accuracy. As expected, the *'without digit-sum'* outperforms the *'with digit-sum'* marginally in the averaged scores, which is consistent with the comparison on the MRPC (whole feature sets). The mean accuracy and F-score of 'Paraphrase Biased' category are (79.3/86.0) for the *'with digit-sum'* and (79.4/86.1) for the *'without digit-sum'*, extrapolating from which, our method is not over-fitting to the MRPC.

In addition, we think this test could be beneficial to the NLP research community for those works addressing PI; e.g. [He et al., 2015] and [Cheng and Kartsaklis, 2015] shown in Table 4 achieve identical accuracy but big discrepancy on F-score is observed. By running the test, robustness of their models can be determined. As for the degree of robustness, priorities of main comparison should be (1) averaged mean accuracy (because averaged distribution is even), (2) averaged mean F-score, (3) averaged standard deviation of accuracy, (4) averaged standard deviation of F-score. Nevertheless, if the two averaged standard deviations are too substantial, e.g. exceeding 3, stability of methods or models should be questioned, despite the fact that high performance on averaged mean accuracy or F-score is achieved.

6.3 Textual Entailment Task (SICK-E)

Our dual weighting scheme seems also confined to binary classification. Therefore, to explore the capability of our method beyond binary classification, we experiment with this multi-classification task.

The SICK-E [Marelli et al., 2014] for relatedness in meaning and entailment is a multi-classification sentence inference task. It contains almost 10,000 sentence pairs (9,840) with three classes included in train/trial/test: entailment (1,274/143/1,404), neutral (2,524/281/2,790), and contradiction (641/71/712). In this work, we use the training and development datasets to tune the classifier.

This Work	**Acc.(%)**
SSSL + FSSSL + SSWPL + FSSWPL	79.29 / 78.80
SSSL + FSSSLs + SSWPL + FSSWPL	82.12 / 82.35
SSSL + FSSSL + SSWPL + FSSWPL (NONOVER)	82.61 / 82.00
SSSL + FSSSLs + SSWPL + FSSWPL (NONOVER)	**84.06 / 83.59**
(Retrofitted) ELMo-PAR[Shi et al., 2019]	**Acc.(%)**
ELMo (all layers)	81.86
ELMo (top layer)	79.64
ELMo-PAR (MRPC)	82.89
ELMo-PAR (Sampled Quora)	81.51
ELMo-PAR (PAN)	83.37
ELMo-PAR (PAN+MRPC+Quora)	**84.46**

Table 7: The table lists the experimental results of testing the SICK-E (accuracy). The results of this work on the **left/right** side are derived from ***'with digit-sum'/'without digit-sum'***.

We introduce two concepts [Yin et al., 2016] to this task: NONOVER and linguistic features. NONOVER denotes removing words occurring in both sentences, and the word-embedding components of empty sentences are 0 vectors in this work. The NONOVER in Table 7 denotes that word-embedding components are derived from NONOVER sentences, but the weights of reward components along with our granularity and flexibility-based features are based on original sentences.

We simplify and then add the linguistic features to FSSSL (FSSSLs). The added features are: (1) absolute difference of negation words counted in S_1 and S_2, negation set in this work is {"no", "not", "aren't", "isn't", "n't", "nobody", "nowhere"}; (2) given S_1, the counts of synonyms, hyponyms, and hypernyms in S_2; (3) given S_1, the counts of antonyms in S_2; (4) sentence length of S_1; (5) sentence length of S_2; (6) sentence length of NONOVER S_1; (7) sentence length of NONOVER S_2. The linguistic features are calculated on NONOVER sentences using WordNet [Miller, 1995].

The assumption of digit-sum works for this task. Compared to the ELMo-PAR [Shi et al., 2019] relying on external paraphrase corpora, our method is more flexible. Based on the pre-trained spaces with dual weighting scheme, performance can be improved with task-specific features. The tuning time on the classifier for best accuracy takes approximately 30 minutes to finish.

7. Conclusion and Future Work

In this paper, we propose a cost-friendly feature-based approach to paraphrase identification. We model sentence similarity by addressing multiple levels of granularity. Fairly competitive performance on the MRPC is achieved compared to task-specific DNNs, and exploration on multi-classification task of SICK-E is also performed. Furthermore, we propose a novel robust test on the QQP to confirm our method is not over-fitting to the MRPC. Our next plan is to test our method on sentence-pair tasks of GLUE benchmark other than the MRPC and QQP.

References

[Bottou, 1998] Bottou, L. (1998). Online learning and stochastic approximations. *On-line learning in neural networks*, 17(9):142.

[Cheng and Kartsaklis, 2015] Cheng, J. and Kartsaklis, D. (2015). Syntax-aware multi-sense word embeddings for deep compositional models of meaning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1531–1542, Lisbon, Portugal. Association for Computational Linguistics.

[Conneau and Kiela, 2018] Conneau, A. and Kiela, D. (2018). SentEval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

[Deerwester et al., 1990] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

[Dolan et al., 2004] Dolan, B., Quirk, C., and Brockett, C. (2004). Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 350–356, Geneva, Switzerland. COLING.

[Fernando and Stevenson, 2008] Fernando, S. and Stevenson, M. (2008). A semantic similarity approach to paraphrase detection. In *Proceedings of the 11th Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics*, pages 45–52.

[Gammerman et al., 1998] Gammerman, A., Vovk, V., and Vapnik, V. (1998). Learning by transduction. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 148–155. Morgan Kaufmann Publishers Inc.

[Graff et al., 2003] Graff, D., Kong, J., Chen, K., and Maeda, K. (2003). English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34.

[Guo and Diab, 2012] Guo, W. and Diab, M. (2012). Modeling sentences in the latent space. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-volume 1*, pages 864–872. Association for Computational Linguistics.

[He et al., 2015] He, H., Gimpel, K., and Lin, J. (2015). Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1576–1586, Lisbon, Portugal. Association for Computational Linguistics.

[Hu et al., 2014] Hu, B., Lu, Z., Li, H., and Chen, Q. (2014). Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*, pages 2042–2050.

[Iyer et al., 2017] Iyer, S., Dandekar, N., and Csernai, K. (2017). First quora dataset release: Question pairs. `https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs`.

[Jaccard, 1912] Jaccard, P. (1912). The distribution of the flora in the alpine zone. 1. *New phytologist*, 11(2):37–50.

[Ji and Eisenstein, 2013] Ji, Y. and Eisenstein, J. (2013). Discriminative improvements to distributional sentence similarity. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 891–896.

[Madnani et al., 2012] Madnani, N., Tetreault, J., and Chodorow, M. (2012). Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 182–190, Montréal, Canada. Association for Computational Linguistics.

[Marelli et al., 2014] Marelli, M., Bentivogli, L., Baroni, M., Bernardi, R., Menini, S., and Zamparelli, R. (2014). SemEval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 1–8, Dublin, Ireland. Association for Computational Linguistics.

[Mikolov et al., 2018] Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., and Joulin, A. (2018). Advances in pre-training distributed word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

[Miller, 1995] Miller, G. A. (1995). Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

[Peters et al., 2018] Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

[Popović, 2015] Popović, M. (2015). chrF: character n-gram f-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

[Shi et al., 2019] Shi, W., Chen, M., Zhou, P., and Chang, K.-W. (2019). Retrofitting contextualized word embeddings with paraphrases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1198–1203, Hong Kong, China. Association for Computational Linguistics.

[Socher et al., 2011] Socher, R., Huang, E. H., Pennin, J., Manning, C. D., and Ng, A. Y. (2011). Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in neural information processing systems*, pages 801–809.

[Tan et al., 2018] Tan, C., Wei, F., Wang, W., Lv, W., and Zhou, M. (2018). Multiway attention networks for modeling sentence pairs. In *IJCAI*, pages 4411–4417.

[Wan et al., 2006] Wan, S., Dras, M., Dale, R., and Paris, C. (2006). Using dependency-based features to take the'para-farce'out of paraphrase. In *Proceedings of the Australasian Language Technology Workshop 2006*, pages 131–138.

[Wang et al., 2018] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2018). GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

[Yang et al., 2019] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Wallach, H., Larochelle, H., Beygelzimer, A., dAlché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 5753–5763. Curran Associates, Inc.

[Yin and Schütze, 2015a] Yin, W. and Schütze, H. (2015a). Convolutional neural network for paraphrase identification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 901–911, Denver, Colorado. Association for Computational Linguistics.

[Yin and Schütze, 2015b] Yin, W. and Schütze, H. (2015b). Discriminative phrase embedding for paraphrase identification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1368–1373, Denver, Colorado. Association for Computational Linguistics.

[Yin et al., 2016] Yin, W., Schütze, H., Xiang, B., and Zhou, B. (2016). ABCNN: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, 4:259–272.