PETCI: A Parallel English Translation Dataset of Chinese Idioms

Kenan Tang
The University of Chicago
kenantang@uchicago.edu

Abstract

Idioms are an important language phenomenon in Chinese, but idiom translation is notoriously hard. Current machine translation models perform poorly on idiom translation, while idioms are sparse in many translation datasets. We present **PETCI**, a parallel English translation dataset of Chinese idioms, aiming to improve idiom translation by both human and machine. The dataset is built by leveraging human and machine effort. Baseline generation models show unsatisfactory abilities to improve translation, but structure-aware classification models show good performance on distinguishing good translations. Furthermore, the size of PETCI can be easily increased without expertise. Overall, PETCI can be helpful to language learners and machine translation systems.

1 Introduction

Idioms are an important yet complex language phenomenon. In certain corpora, 3 out of 10 sentences are estimated to contain idioms (Moon, 1998). The pervasively usage of idioms entails a wide range of linguistic studies (Cacciari and Tabossi, 2014). In NLP, idioms are involved in various tasks, including idiom recognition (Peng and Feldman, 2016; Liu and Hwa, 2018), embedding learning (Tan and Jiang, 2021), and idiom comprehension (Zheng et al., 2019).

However, these NLP systems assume a monolingual setting, and the models perform poorly due to the non-compositional nature of idioms. Adopting a multilingual setting may be beneficial, but existing multilingual idiom datasets are rather small (Moussallem et al., 2018). Machine translation systems also cannot help, because idiom translation is challenging (Salton et al., 2014; Cap et al., 2015; Fadaee et al., 2018). Idiom translation is even difficult for human, as it requires knowing both the source and target culture and mastering diverse translation strategies (Wang and Wang, 2013).

Despite these difficulties, there are not enough datasets for model improvement. Translating idioms from scratch is very laborious. Nevertheless, existing idiom dictionaries have always been readily available. Moreover, they usually provide more than one translation to each idiom, which naturally results in a paraphrasal dataset. The paraphrase strategies used for idiom translation are usually more aggressive than in other paraphrasal datasets such as PPDB (Ganitkevitch et al., 2013). Therefore, neural network models based on these dictionaries may benefit idiom translation more than naive retrieval.

To this end, we present **PETCI**, a **P**arallel **E**nglish **T**ranslation dataset of **C**hinese **I**dioms. PETCI is collected from an idiom dictionary, together with machine translation results from Google and DeepL. Instead of directly using models to translate idioms, we design tasks that require the model to distinguish gold translations from unsatisfactory ones and to rewrite translations to improve their quality. We show that models perform better with the increase of the dataset size, while such increase requires no linguistic expertise. PETCI is publicly available[1].

The main contribution of this paper is two-fold. First, we collect a dataset that aims improve Chinese idiom translation by both machine translation systems and language learners. Secondly, we test several baseline models on the task we defined.

In the following sections, we begin by a detailed description of our data collection process (Section 2), followed by case studies and statistics (Section 3). Based on the dataset, we define tasks and report the performance of baseline models we choose (Section 4, 5, 6). Then, we discuss our observations, provide relevant work and future directions (Section 7), and conclude (Section 8).

[1]https://github.com/kt2k01/petci

Chinese	Dictionary Translation	Machine Translation	Issues Identified
为虎傅翼	• assist an evil-doer • give wings to a tiger • increase the power of a tyrant to do evil • lend support to an evil-doer like adding wings to a tiger	• Fu Yi for the tiger • for the tiger's wings • for the tiger • for the tiger's wing • for the tiger's sake	Name hallucination Single-word edit Very long alternative
煮豆燃萁	• fratricidal strife • burn beanstalks to cook beans - one member of a family injuring another • boil beans with beanstalks - reference to a fight among brothers	• boiled beans • burning beanstalks cook the beans (idiom); to cause internecine strife	Partial translation Hyphenation Semicolon
风流云散	• (of old companions) separated and scattered • blown apart by the wind and scattered like clouds • (of relatives or friends, separated from each other) as the wind blowing and the clouds scattering	• wind and clouds • Wind flow and clouds scatter • Wind flow and clouds scattered	Parenthesis Literal translation Explicit parallelism
萧规曹随	• follow established rules • Tsao (a Han Dynasty prime minister) followed the rules set by Hsiao (his predecessor)	• Xiao Gui Cao Sui • lit. Xiao Gui Cao follows the rules (idiom); fig. follow the rules of a certain place	Different romanization Unstable reference

Table 1: Examples of idioms and translations in PETCI. Parts of the translations are colored and underlined, if they relate to the identified issues (details in Section 2.3). The first Machine translation is always from Google.

2 Data Collection

2.1 Human Translation

We collect human translations from a English translation dictionary of Chinese idioms (Liu, 1989). This dictionary is preferred because it provides multiple translations to a single idiom, with one of them labelled as being most frequently used. Though the dictionary is published 3 decades ago, the time is short compared to the observed long time for the meaning of idioms to change (Nunberg et al., 1994).

We use the OCR tool `tesseract`[2] to extract the English translations from a scanned version of the book. The Chinese idioms are then manually added. We performed both automatic and manual check to ensure the quality of final results (details in Appendix A).

[2]https://tesseract-ocr.github.io/

2.2 Machine Translation

We collect machine translations from the publicly available models of Google and DeepL[3]. Though they both provide free APIs for developers, richer information is returned from direct queries on the website interface (details in Appendix B). For example, while Google always returns a single translation, DeepL sometimes returns alternative translations.

[3]We do not use the Baidu translation that has better performance on Chinese, because it directly returns a dictionary translation for most of the time.

We decide to use the alternative translations. Therefore, instead of using developer APIs, we have to scrape translations from the web interface. Because the shadow DOM prevents the content from being scraped by `selenium`, we use a script that automatically paste our source idiom into the source box, and take a screenshot of the result box. The `tesseract` OCR tool is then used on the screenshots to retrieve the text. Though OCR performs much better on screenshots than on scans as expected, we manually check the results to ensure quality.

2.3 Case Study

Here, we summarize our observations during data collection. Examples are listed in Table 1. In order to avoid confusion, for the rest of this paper, we use **Gold** translations to refer to the set of the first translation of all idioms in the **Dictionary** translations. The rest of the Dictionary translation is denoted as **Human** translations. We use **Machine** translations to refer the combination of **Google** and **DeepL** translations. In other words, we have:

- Dictionary $=$ Gold $\cup$ Human
- Machine $=$ Google $\cup$ DeepL

Non-uniform Numbers of Alternatives Alternative translations are provided by both the Dictionary and DeepL. The number of alternatives from the Dictionary varies from 0 to 13. DeepL provides only up to 3 alternatives, but the first translation sometimes contains two translations separated by a semicolon.

Low Edit Distance The multiple alternatives from DeepL usually result from single-word edits, when the translation itself is much longer than one word. This almost never happens in the Dictionary.

Dictionary Artifacts The Dictionary translations aim at human readers, so part of the translation may only be explanatory. Indicators of explanation include parenthesis, dash, and abbreviations such as *sb.* (somebody) and *sth.* (something). We expand the abbreviations but keep the other explanations. Also, due to the time it was published, the Dictionary uses the Wade-Giles romanization system, which is different from the Hanyu Pinyin used by Machine.

Machine Artifacts Some Machine translations can be the same as Human or even Gold ones. However, mistakes do exist. We notice meaningless repetition and spurious uppercase letters. In some cases, the uppercase letters seems to result from the machine recognizing part of an idioms as names. In very rare cases, Machine translations contain untranslated Chinese characters. We substitute them by the Hanyu Pinyin. All these artifacts are unique to Machine translations.

Unstable References DeepL sometimes refers to other dictionaries. However, DeepL may replace part of the correct dictionary translation by machine artifacts. Also, DeepL uses abbreviations inconsistently, indicating that it refers to multiple dictionaries. Despite the lack of citation information, we identify one of the dictionaries to be *4327 Chinese idioms* (Akenos, 2012), which uses the two abbreviations *fig.* and *lit.*.

Literal Machine Translation The literal Machine translation is even less satisfactory than the literal Human translation for two reasons. First, Machine sometimes returns a partial translation of only 2 out of the 4 Chinese characters. Secondly, Machine may translate the characters one by one, ignoring the non-compositionality of idioms.

Explicit Structure Parallel structures are pervasive in Chinese idioms (Wu, 1995) but usually not explicitly indicated by a conjunction character. Parallelism can be revealed in the English translation by conjunctions. However, parallel structures are sometimes considered redundant, and abridging is used to remove parallelism in translations of Chinese idioms without the loss in meaning (Wang and Wang, 2013). Structures other than parallelism are discussed in (Wang and Yu, 2010), but they cannot be naively detected from translations.

Optimality of Gold Translations Many Human translations are more verbose than the Gold translation. This is consistent with the tendency of language learners to err on the verbose and literal side. However, many Human alternatives are also much more succinct than the Gold translation, which can be arguably better. Without further linguistic analysis, we accept the Gold translations as labelled by the dictionary.

3 Statistics

In this section, we compute statistics of PETCI, and compare with the existing idiom datasets. We then use our dataset to probe the appearance of idioms in other translation datasets. We also provide a quantitative analysis for some our observations in Section 2.3.

Dataset	Chinese	English
Oxford	—	10,000
Xinhua	10,481	—
4327	4,327	8,654
CIBB	50	100
CIKB	38,117	33K - 114K
PETCI	4,310	29,936

Table 2: The comparison between the sizes of dictionaries and datasets. Some sizes are estimated.

Dataset	Size (M)	PoI (%)
ParaCrawl v7.1	14.17	1.56
News Commentary v16	0.32	7.70
Wiki Titles v3	0.92	0.02
WikiMatrix	2.60	0.90
Back-translated news	19.76	1.75

Table 3: The Percentage of sentences using Chinese Idioms (PoI) of different datasets from the WMT21 news translation task.

3.1 Comparison with Existing Datasets

We summarize our comparison of dataset sizes in Table 2. Monolingual idiom dictionaries in both English and Chinese are usually much larger than bilingual idiom dictionaries. For comparison, we choose *Oxford Dictionary of Idioms* (Ayto, 2020), *Xinhua Idiom Dictionary* (Zhao et al., 2015), and *4327 Chinese idioms* (Akenos, 2012). The precise number of translations in *4327 Chinese idioms* is not available, but the dictionary usually provides both a literal and a figurative translation. Therefore, we estimate the number of translations to be twice the number of Chinese idioms.

There have also been attempts to build a dataset of English translated Chinese idioms. For example, the CIBB dataset contains the English translation of Chinese idiom, together with a blacklist of words that indicate unwanted literal translation (Shao et al., 2018). This dataset has been used to score translation models (Huang et al., 2021), but it has an extremely small size of only 50 distinct Chinese idioms and a pair of translations for each idiom. Another dataset, CIKB, thoroughly contains 38,117 idioms with multiple properties (Wang and Yu, 2010). Among all properties, each idiom has at most 3 English translations, including the literal translation, free translation, and the English equivalent. However, of the 38,117 idioms, only 11,000 have complete properties (among which 3,000 are labelled as most commonly used). Though the CIKB dataset is not made public, we reasonably assume that the missing properties are mainly English translations, because the other properties are in Chinese and thus easier to obtain. We estimate the number of English translations in CIKB to be between 3 times the number of complete entries and of all entries.

From our comparison, we see that PETCI covers a large percentage of frequently used Chinese idioms, while uniquely providing much more parallel translations. To further grow the size of PETCI, we propose 3 orthogonal directions: (1) increase the number of idioms, (2) add more translations to each idiom, and (3) add multiple gold translations to each idiom. The first two directions can be accomplished by language learners without linguistic expertise. The third direction, though requiring expertise, is revealed to be useful by the model performance.

3.2 Percentage of Chinese Idioms

We would like to know how many sentences in widely used Zh-En datasets contain at least one idiom from our dataset. We consider the datasets from the news translation task from WMT21, with intuitively the highest percentage of idiom usage among all provided domains. The result in Table 3 shows that all the datasets have a low Percentage of sentences using Chinese Idioms (PoI), with the PoI of the two datasets related to Wikipedia lower than 1%.

The low PoI is not only observed in the Zh-En pair (Fadaee et al., 2018). Also, the PoI might differ in the En-Zh and Zh-En directions, though the available Zh-En datasets from WMT are all undirectional.

3.3 Quantitative Case Study

In this subsection, we qualitatively analyze our observations in the case study (Section 2.3). The results are summarized in Table 4.

Length The average length of Chinese idioms in PETCI is 4.30, with 91.42% of all idioms having 4 characters. We tokenize translations using the `stanza` package (Qi et al., 2020), and report the average length of translations in tokens and characters. Surprisingly, the length of Human translations and Gold translations are extremely close. By calculating the percentage of different translations that are longer or shorter than the gold translation,

Statistics	Dictionary	Gold	Human	Machine	Google	DeepL
Total number	14997	4310	10687	14939	4310	10629
Avg. length (token)	5.03	4.97	5.06	3.72	2.81	4.09
Avg. length (char)	27.14	27.15	27.14	19.99	15.72	21.72
% longer (token)	—	—	47.33	25.44	11.39	34.14
% shorter (token)	—	—	33.16	54.75	71.07	48.13
% longer (char)	—	—	56.04	30.05	14.80	36.23
% shorter (char)	—	—	40.57	63.82	78.52	57.86
% with NNP	3.67	0.77	2.99	4.43	2.99	2.09
% with CC	39.72	18.26	31.00	32.44	9.84	30.32
Avg. edit distance	5.24	—	—	—	—	2.45
% of single edit	0.83	—	—	—	—	27.20

Table 4: The statistics of different subsets in PETCI. *Longer* and *shorter* are based on comparison with the Gold translations, therefore not applicable to Dictionary translations. *Edit distances* are only available for subsets that contain multiple translations. Human is a subset of Dictionary, so its average edit distance is not calculated.

we see that long sub-optimal translations are still unique to Human translations. The Machine translations tend to be shorter due to partial translation.

Name Hallucination We would like to know how much hallucinated names has been introduced in the Machine translations. We used `stanza` to perform part-of-speech (POS) tagging on the translations, and calculate the percentage of idioms that has at least one NNP token in its translation. The Machine translations have a NNP percentage much higher than the Gold translation, but close to the Human translations. This may result from that Human translations include explanations that involve names in a story from which a idiom originates.

Parallel Structure The popular Chinese word segmentation package `jieba` considers 93.64% of the Chinese idioms in PETCI as an atomic unit. However, the idioms obviously have internal structures. For a coarse estimation, we consider a translation containing the CC token to have parallel structure. We calculate the percentage of idioms that has at least one CC token in its translations. We see that parallel structure is pervasive in PETCI, but less favored by the Gold translation. Also, the DeepL translations can catch parallel structures as well as the Human translations. Google performs less well.

Edit Distance To quantitatively analyze the simple-word edit strategy used by DeepL, we calculate the average token-level Levenshtein distance of all pairs of translation for the same idiom. We then calculate the percentage of single-word edit pairs, with the length of translation being at least 2 tokens. The same calculation is also applied to Dictionary translations. We notice a lower edit distance and higher percentage of single-word edits from DeepL, which is consistent with our qualitative observation. This suggests the possibility of further augmenting the data by single-word edit methods, such as replacing synonyms based on WordNet (Miller, 1992).

Split and Filter We observe that DeepL may join multiple translations into one, and that all translations may sometimes include dictionary artifacts. Therefore, all statistics is performed after the removal of 2,134 parenthesized content and 341 abbreviations. To further prepare the dataset for training, we split 250 translations that include a semicolon. After the split, we remove translations from the Machine set that also appeared in the Dictionary. The processed Human/Machine translation set has a size of 10,690/13,539. In the following sections, we have:

- Machine $\leftarrow$ $Split$(Machine) $\setminus$ Dictionary

Explicitly incorporating the above translation features into models may improve their performance, but this strategy is outside the scope of this paper. For all models, we only use the text as the input.

4 Models

A human expert is able to tell whether a translation is satisfactory, and then rewrite the translation if it is not. We expect the models to behave similarly. Therefore, we design a binary classification task and a rewriting task. The binary classifier should determine whether a given translation is Gold or not. The rewriting model should rewrite other translations in the standard of Gold ones. The two tasks may be combined in a pipeline, but we leave that for future work, and use the following models to perform either of the two tasks. No major changes are made on the model architectures, so we omit detailed descriptions and refer readers to the original papers.

4.1 LSTM Classification

The recurrent neural network (RNN) with LSTM cells (Hochreiter and Schmidhuber, 1997) is able to encode a sentence of variable length into a hidden state vector. Then, the vector can be fed into a linear layer for classification.

Some variants of LSTM, such as Bidirectional LSTM and Multilayer LSTM (Graves et al., 2013), can perform better on longer sequences, but most translations in PETCI are short. Moreover, we would like to introduce fewer variables when comparing models, so we only use the one-directional single-layer LSTM.

4.2 Tree-LSTM Classfication

We have shown that the translation of Chinese idioms exhibits special structures that may be used to improve model performance. A variant of LSTM that explicitly accounts for tree structures is the Tree-LSTM (Tai et al., 2015).

The attention mechanism can be further incorporated into the Tree-LSTM (Ahmed et al., 2019), but we do not use it for the sake of comparison.

4.3 BERT Classification

The BERT model has been reported to have superior performance on sentence classification tasks (Devlin et al., 2019). It can also capture the structural information of the input text (Jawahar et al., 2019).

We do not use better variants of BERT for the following considerations. First, some variants are pre-trained on different datasets other than the BookCorpus (Zhu et al., 2015) and English Wikipedia used by BERT. RoBERTa, for example, has part of its pre-training data from Reddit and CommonCrawl (Liu et al., 2019). Intuitively, idioms are used more in the colloquial language of Reddit. Though we are not able to estimate the English PoI in these datasets, we do see that WikiMatrix has a much lower PoI than ParaCrawl. Similarly, we expect the Wikipedia dataset to have a lower PoI than CommonCrawl. The choice of pre-training datasets by RoBerta may benefit our downstream task, but not ideal for fair comparison.

Secondly, other variants of BERT may improve the sematic representation of phrases and sentences, such as PhraseBERT (Wang et al., 2021) and SentenceBERT (Reimers and Gurevych, 2019). However, they are also not ideal, because idiom translation covers a large range of text length. A user of the model may prefer to give either over-simplified (like Machine) or over-extended (like Human) translations, and the model is expected to perform well in both cases. Therefore, we take the vanilla BERT as a middle ground.

4.4 OpenNMT Rewriting

Rewriting is a sequence-to-sequence task. We use the OpenNMT framework which provides an encoder-decoder model based on LSTM with the attention mechanism (Klein et al., 2017). A part of the rewriting task can be viewed as simplification, on which the OpenNMT framework is reported to perform well (Nisioi et al., 2017).

We do not consider the SOTA sequence-to-sequence models, such as BART (Lewis et al., 2020a) or T5 (Raffel et al., 2019), because they are much larger and forbiddingly difficult for language learners to use.

5 Experiments

5.1 Classification

For the binary classification, we use train/dev/test splits of 3448/431/431 idioms. To study the effect of blending Human and Machine translation, we construct 3 types of training sets that contain (1) both Gold and Human (H), (2) both Gold and Machine (M), and (3) all Gold, Human, and Machine translations (HM). The development sets are similarly constructed and used when training on the corresponding set. The test set is the combination of Gold, Human, and Machine translations, meaning that models trained on the HM set will be evaluated on Machine translations for zero-shot accuracy. To balance the training data, whenever we add a Human or Machine translation into the training set, we always add its corresponding Gold translation once. The development and test sets are not balanced.

We also want to examine whether increasing the size of the training set may improve model performance. We construct partial training sets from the full set. In the case of training set H, the smallest partial set is constructed by taking exactly one Human translation from each idiom in the training set, together with Gold translations. The size of 3 other partial sets are then evenly spaced between the sizes of the smallest and the full training set, resulting in 5 H sets in total. For the 3 partial sets with middle sizes, we randomly sample from the remaining Human translations to reach the target size. The sizes of 5 H training sets are 3,050/4,441/5,833/7,224/8,616, expressed in the number of non-Gold translations (the total size is strictly twice as large). We similarly construct 5 M sets (sizes 3,419/5,265/7,112/8,958/10,805), and merge the H and M sets to create 5 HM sets. Here, distributional shift is inevitably introduced, but this process mimics the real-world scenario where it is impossible for a model user to add extra Gold translations, but can only add their own non-Gold translations of an unknown distribution compared to the existing H and M sets.

For both LSTM and Tree-LSTM, we use the cased 300-dimensional GloVe vectors pre-trained on CommonCrawl (Pennington et al., 2014) to initialize word representations. We allow the word representations to be updated during training to improve performance (Tai et al., 2015).

We use the binary constituency Tree-LSTM because it is reported to out-perform the dependency Tree-LSTM. We use CoreNLP to obtain the constituency parse trees for sentences (Manning et al., 2014). The binary parse option (not available in `stanza`) is set to true to return the internal binary parse trees. Half-nodes with only 1 child are removed.

However, fine-grained annotation on each node is not available, so only full trees are used during training. All nodes other than the root have a dummy label 0. We make this modification on a PyTorch re-implementation of the Tree-LSTM [4].

We implement BERT classification using HuggingFace (Wolf et al., 2019). The pre-trained model we use is the `bert-base-uncased`.

5.2 Rewriting

We use all Human and Machine translations for the idioms in the train split as the source training set. The corresponding Gold translations are the target. The development and test sets are similarly constructed from the dev/test splits. We do not change the training set size for rewriting, and this choice will be justified when we discuss the results.

Our model architecture is the same as the NTS model based on OpenNMT framework (Nisioi et al., 2017). Though originally designed to perform text simplification, the NTS model has no component that explicitly performs simplification. We only change pre-trained embeddings from word2vec (Mikolov et al., 2013) to GloVe.

5.3 Hyperparameters and Training Details

To ensure fair comparison, the hyperparameters for each model are fixed for different training sets. For LSTM and Tree-LSTM, we follow the choice of (Tai et al., 2015). For BERT, (Devlin et al., 2019) suggest that a range of parameters work well for the task, so we choose a batch size of 16, a learning rate of 5e-5, and an epoch number of 3. For OpenNMT, we follow the configuration provided by (Nisioi et al., 2017). More details can be found in Appendix C.

6 Results and Discussion

6.1 Classification

The results are plotted in Figure 1, with numeric results in Appendix D. The suffix of a model indicate the training set that it is trained on.

Gold Sparsity Due to the sparsity of gold translations, a model may blindly assign the gold label with low probability, and the accuracy is still high. Therefore, we compare all results to a random baseline that assigns the label based on the probability equal to the label frequency in PETCI. The accuracy of the random baseline is 75.35% overall, 14.40% on Gold, and 85.60% on either Human or Machine. Most models underperform the random baseline. Also, the Gold accuracy of all models deteriorates with the increase in the training set size, but still above the random baseline. Our data balancing strategy does not seem to overcome the sparsity problem.

Training Set Size With the increase in training set size, the overall accuracy mostly increases. The increase in performance has not saturated, indicating the need of further increasing dataset size. The BERT-HM model trained on the most data performs best, closely followed by Tree-LSTM-HM. When trained on the most data, LSTM does not out-perform the random baseline. In few cases, the increase in dataset size hurts performance, such

[4] `https://github.com/dmlc/dgl/tree/master/examples/pytorch/tree_lstm`

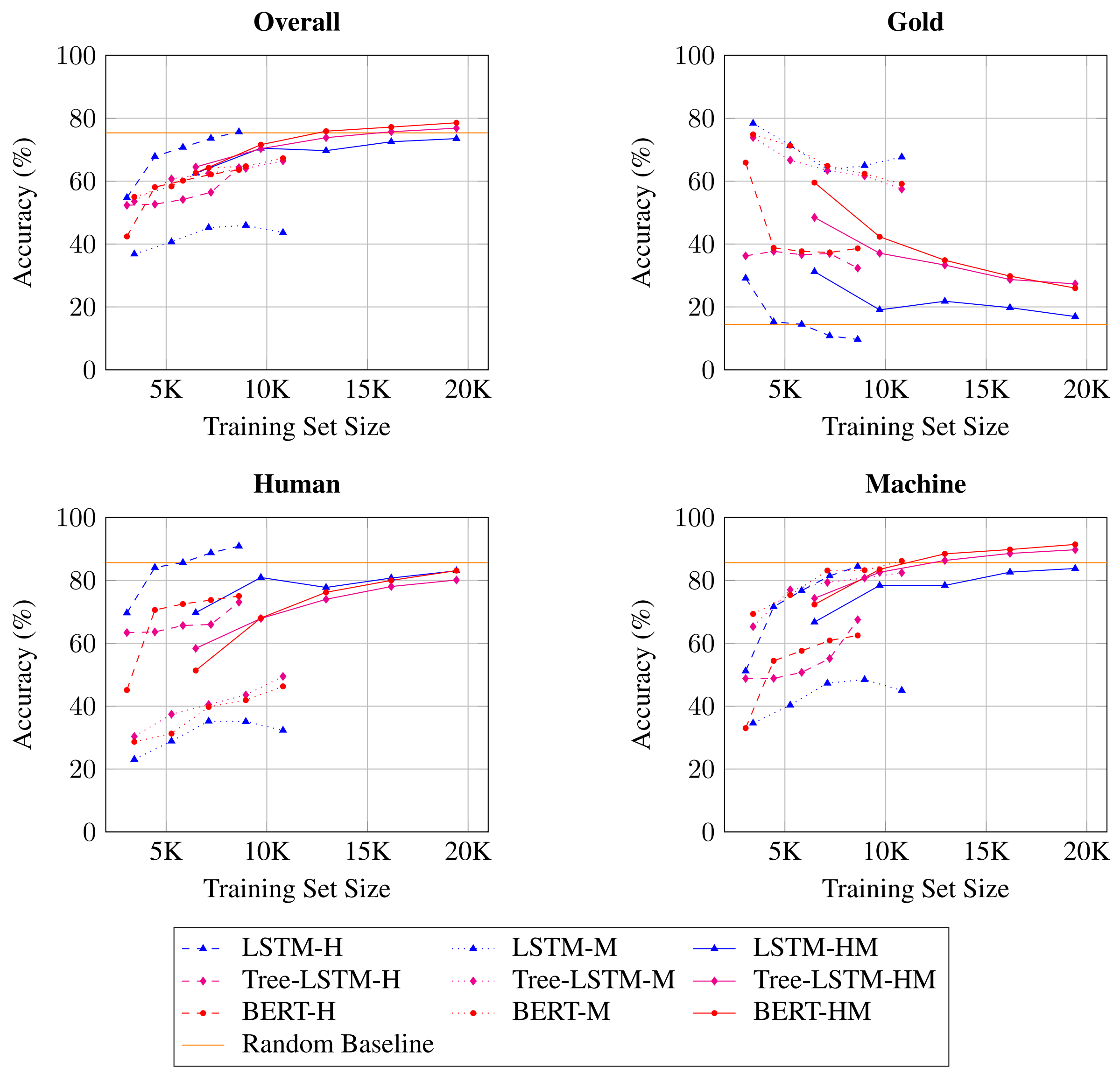


Figure 1: Test accuracies of models on different subsets. The suffix of a model indicates the training set it is trained on. We report mean accuracies over 5 runs.

as for LSTM-M. This may be an artifact of high standard deviation.

Human and Machine The models trained either on H or M sets have a low zero-shot accuracy on the opposite side, while training on the merged HM sets benefits both. The LSTM-H model seems to a high accuracy comparable to the random baseline, even outperforming LSTM-M on Machine without ever seeing a Machine translation. However, this seems to be an artifact of over-assigning the non-Gold label. The good performance of -HM models indicates that it is reasonable to combine Machine and Human translation in the training set, even when the model is responsible for classifying translation that only comes from either MT systems or language learners.

Detecting Structures The subset accuracy reflects the structural difference of translations and a model's ability to detect it. The highest Gold accuracy come from the -M models, probably because it is easier for the model to distinguish Gold and Machine translations due to the broken structures of machine translation, indicated by the high NNP percentage and low CC percentage. LSTM-M has an unexpected low performance on the Machine set, maybe due to its inability to detect structure. In contrast, the Human translations resemble Gold translations in structure more than Machine translations, which may confuse the -H and -HM models.

Confusion from Structures We also notice the interesting fact that the Gold accuracy BERT deteriorates more rapidly than Tree-LSTM. This may be because that the Tree-LSTM model is better at explicitly identifying good structures that adds credibility to Gold translations. In the Human set, however, the identification of structure confuses both Tree-LSTM and BERT, preventing them from out-performing the sequential LSTM that is unaware of the structure. The Machine set creates no such confusion.

A detailed analysis of the failure cases would require linguistic knowledge, which is beyond the scope of this paper. Coarse-grained metrics such as the length or the POS percentage are not sufficient.

6.2 Rewriting

The good performance of -HM models when trained on the full set justified our choice of training set for the rewriting task. The decreasing accuracy on Gold translations has less effect on the rewriting task, since we do not rewrite Gold translations.

Despite the success of classification models, the training of the rewriting model failed. During training, though the perplexity on the training set is reduced, the perplexity on the development set remains high, indicating an inevitable over-fitting. The performance on the test set is visibly poor, defying the need for calculating automatic metrics.

We believe that OpenNMT fails because the rewriting task is too aggressive. The average token-level Levenshtein distance from Human/Machine to Gold is 5.39/4.93, close to the average Gold token length 4.97. Our situation is very different from the traditional sentence simplification tasks, where strategies are much more conservative and the overlapping rate is high (Zhang and Lapata, 2017). To rewrite, the model even has too guess from partial translations. The parametric knowledge in GloVe vectors does not suffice, and non-parametric knowledge should be incorporated by methods such as retrieval-augmented generation (Lewis et al., 2020b).

7 Related Work

Leveraging Human Translation There are works that leverage existing human translations, either from experts (Shimizu et al., 2014) or language learners (Dmitrieva and Tiedemann, 2021). The collected datasets are innately more varied than the parallel corpus collected by back-translation or other automated methods, such as in (Kim et al., 2021). In our work, we try to use both human and machine translations to increase the size of PETCI, and they cooperate well.

Faster BERT Ideally, after expanding the dataset, the model needs to be re-trained. The sheer size of Transformer-based models may prohibit language learners from using it. Smaller versions of BERT have been proposed, such as DistilBERT (Sanh et al., 2020). However, the 40% decrease in model size comes at a cost of 3% decrease in performance, which may cause BERT to lose its slight advantage over smaller models such as Tree-LSTM. Similarly, quantization and pruning compress the BERT model at a factor of no more than 10, but at a cost of 2% decrease in performance (Cheong and Daniel, 2019).

Evaluation Metrics Our rewriting task is close to sentence simplification. Metrics such as BLEU (Papineni et al., 2002) can be used to evaluate sequence generation quality, but are shown to be less correlated with human judgements on simplification quality than another metric SARI (Xu et al., 2016). However, to establish the correlation between any of these metrics with PETCI rewriting quality, we need fine-grained labels. The Flesch-Kincaid readability score can also be used to judge the readability of translation, but it is not guaranteed to correlate with the ease of understanding idiom translations. The cosine similarity between translations in PETCI is not calculated, but a low similarity is expected due to the pervasive metaphoric language in idioms.

8 Conclusion

In this paper, we build a Parallel English Translation dataset on Chinese Idioms (PETCI) and show its advantages over the existing Chinese idiom datasets. Based on PETCI, we propose a binary classification and a rewriting task, and the baseline models we choose succeed on the former, meanwhile showing great potential when the dataset size increases. Also, we build the datasets and choose the models to make them accessible to language learners, which allows further improvement without the involvement of translation experts.

Acknowledgements

The author thanks Natsume Shiki for insightful comments and kind support.

References

Mahtab Ahmed, Muhammad Rifayat Samee, and Robert E. Mercer. 2019. Improving tree-lstm with tree attention. In *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, pages 247–254.

Aris Akenos. 2012. *4327 Chinese idioms*. Lulu.com.

J. Ayto. 2020. *Oxford Dictionary of Idioms*. Oxford paperback reference. Oxford University Press.

Cristina Cacciari and Patrizia Tabossi. 2014. *Idioms: Processing, structure, and interpretation*. Psychology Press.

Fabienne Cap, Manju Nirmal, Marion Weller, and Sabine Schulte im Walde. 2015. How to account for idiomatic German support verb constructions in statistical machine translation. In *Proceedings of the 11th Workshop on Multiword Expressions*, pages 19–28, Denver, Colorado. Association for Computational Linguistics.

Robin Cheong and Robel Daniel. 2019. transformers. zip: Compressing transformers with pruning and quantization. *Technical report, tech. rep., Stanford University, Stanford, California*.

Michael Cooper and Matthew Shardlow. 2020. CombiNMT: An exploration into neural text simplification models. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5588–5594, Marseille, France. European Language Resources Association.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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.

Anna Dmitrieva and Jörg Tiedemann. 2021. Creating an aligned Russian text simplification dataset from language learner data. In *Proceedings of the 8th Workshop on Balto-Slavic Natural Language Processing*, pages 73–79, Kiyv, Ukraine. Association for Computational Linguistics.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2018. Examining the tip of the iceberg: A data set for idiom translation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia. Association for Computational Linguistics.

Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. 2013. Hybrid speech recognition with deep bidirectional lstm. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 273–278.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Dandan Huang, Kun Wang, and Yue Zhang. 2021. A comparison between pre-training and large-scale back-translation for neural machine translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1718–1732, Online. Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Joongwon Kim, Mounica Maddela, Reno Kriz, Wei Xu, and Chris Callison-Burch. 2021. BiSECT: Learning to split and rephrase sentences with bitexts. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6193–6209, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Changsheng Liu and Rebecca Hwa. 2018. Heuristically informed unsupervised idiom usage recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1723–1731, Brussels, Belgium. Association for Computational Linguistics.

Yi Liu. 1989. *Zhong Wen Cheng Yu Ying Yi Ci Dian = a dictionary of Chinese idioms with English translation*. Learning Publishing.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at International Conference on Learning Representations*.

George A. Miller. 1992. WordNet: A lexical database for English. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

Rosamund Moon. 1998. *Fixed expressions and idioms in English: A corpus-based approach*. Oxford University Press.

Diego Moussallem, Mohamed Ahmed Sherif, Diego Esteves, Marcos Zampieri, and Axel-Cyrille Ngonga Ngomo. 2018. LIdioms: A multilingual linked idioms data set. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Sergiu Nisioi, Sanja Štajner, Simone Paolo Ponzetto, and Liviu P. Dinu. 2017. Exploring neural text simplification models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 85–91, Vancouver, Canada. Association for Computational Linguistics.

Geoffrey Nunberg, Ivan A Sag, and Thomas Wasow. 1994. Idioms. *Language*, 70(3):491–538.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Jing Peng and Anna Feldman. 2016. Experiments in idiom recognition. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2752–2761, Osaka, Japan. The COLING 2016 Organizing Committee.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Giancarlo Salton, Robert Ross, and John Kelleher. 2014. Evaluation of a substitution method for idiom transformation in statistical machine translation. In *Proceedings of the 10th Workshop on Multiword Expressions (MWE)*, pages 38–42, Gothenburg, Sweden. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Yutong Shao, Rico Sennrich, Bonnie Webber, and Federico Fancellu. 2018. Evaluating machine translation performance on Chinese idioms with a blacklist method. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Hiroaki Shimizu, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2014. Collection of a simultaneous translation corpus for comparative analysis. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 670–673, Reykjavik, Iceland. European Language Resources Association (ELRA).

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In

Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1556–1566, Beijing, China. Association for Computational Linguistics.

Minghuan Tan and Jing Jiang. 2021. Learning and evaluating Chinese idiom embeddings. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 1387–1396, Held Online. INCOMA Ltd.

Lanchun Wang and Shuo Wang. 2013. A study of idiom translation strategies between english and chinese. *Theory and practice in language studies*, 3(9):1691.

Lei Wang and Shiwen Yu. 2010. Construction of Chinese idiom knowledge-base and its applications. In *Proceedings of the 2010 Workshop on Multiword Expressions: from Theory to Applications*, pages 11–18, Beijing, China. Coling 2010 Organizing Committee.

Shufan Wang, Laure Thompson, and Mohit Iyyer. 2021. Phrase-BERT: Improved phrase embeddings from BERT with an application to corpus exploration. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10837–10851, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Chu-hsia Wu. 1995. On the cultural traits of chinese idioms. *Intercultural Communication Studies*, 5:61–82.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594, Copenhagen, Denmark. Association for Computational Linguistics.

Keqin Zhao, Wanqi Zhang, Zhensheng Xu, Wanping He, Yu Wang, and Xinxin Jin. 2015. *Xinhua Idiom Dictionary*. The Commercial Press.

Chujie Zheng, Minlie Huang, and Aixin Sun. 2019. ChID: A large-scale Chinese IDiom dataset for cloze test. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 778–787, Florence, Italy. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27.

A Data Cleaning Details

It takes around 50 hours to manually clean up and add the Chinese idioms into the OCR results. Special characters replace alphabetic characters in the OCR results of the dictionary, such as @ for a. Also, letters with similar shape are sometimes confused, such as {a, e, o, s}, {i, l, t}, and {r, v}. The results are then checked by the spelling correction provided by the Mac TextEdit. The example sentences are removed.

B Web Interface

The screenshots of Google and DeepL translation web interface are shown in Figure 2. The translation data is collected on February 5 and 6 for DeepL and Google, and he screenshots are taken on February 17, 2022. The results are the same across the short time span. Extra information returned from the translation model, such as dictionary translations (shown in the red box), are not used in this paper because they are neither easily scraped nor consistently returned. Also, the alternative results from DeepL are not returned when we put more than one line of text in the source box.

C Further Training Details

In this section, we list the hyperparameters we have used for all models. If not listed, the hyperparamters are set to the default values provided by the packages.

Hyperparameters for LSTM are:

- memory dimension: 168
- batch size: 25
- dropout rate: 0.5
- optimizer: AdaGrad
- learning rate: 0.05

Hyperparameters for Tree-LSTM are:

- memory dimension: 150
- batch size: 25
- dropout rate: 0.5
- optimizer: AdaGrad
- learning rate: 0.05

For LSTM and Tree-LSTM, we also try a batch size of 20, but observe no qualitatively different results.

Hyperparameters for BERT are:

Model	$\|\theta\|$
LSTM	316k
Tree-LSTM	316k
BERT	110M
OpenNMT	23M

Table 5: Number of parameters.

- batch size: 16
- epochs: 3
- save steps: 500
- warmup steps: 500
- optimizer: AdamW
- learning rate: 5e-5
- weight decay: 0.01

Hyperparameters for OpenNMT are:

- LSTM layers: 2
- hidden state size: 500
- hidden units: 500
- dropout probablity: 0.3
- save steps: 300
- optimizer: SGD
- step when learning rate is halved: 2400
- learning rate: 1

For OpenNMT, we follow the advice of (Cooper and Shardlow, 2020) to convert the number of epochs to steps when re-implementing the model from (Nisioi et al., 2017). We also try to change the optimizer to AdaGrad or Adam with the recommended learning rate, but the training does not succeed.

For LSTM and Tree-LSTM, we apply early stopping when the accuracy does not increase on the development set for 10 consecutive epochs. For BERT, we choose from the last 5 checkpoints the checkpoint that has the best accuracy on the development set. For OpenNMT, we apply early stopping when perplexity has not improved on the development set for 5 checkpoints.

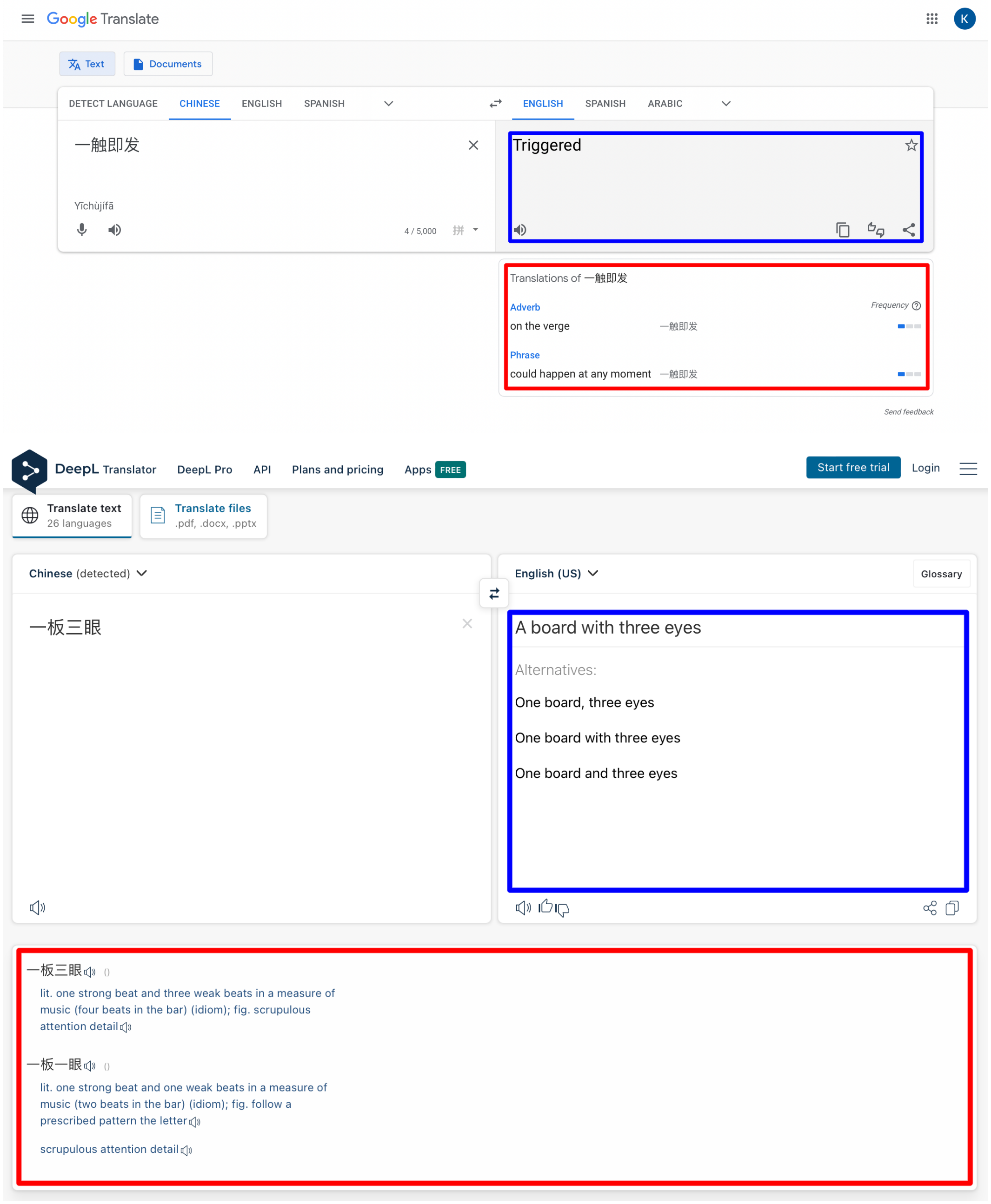
Due to the drastic differences in the number of parameters (Table 5), the LSTM and Tree-LSTM model are trained on CPU, while the BERT and OpenNMT model are trained on 1 GPU assigned by Google Colab.

D Numeric Results

Table 6 shows the numeric results of Figure 1.

Test Set	Model	Training Set Size				
		1	2	3	4	5
Overall	LSTM-H	54.75 (1.92)	67.84 (3.26)	70.75 (3.85)	73.61 (2.32)	75.66 (1.75)
	LSTM-M	36.83 (1.93)	40.67 (1.54)	45.21 (1.83)	45.93 (2.73)	43.65 (3.57)
	LSTM-HM	62.55 (1.20)	70.47 (2.18)	69.69 (3.09)	72.52 (1.14)	73.53 (2.05)
	Tree-LSTM-H	52.34 (5.31)	52.66 (6.06)	54.16 (1.97)	56.45 (0.55)	64.31 (2.10)
	Tree-LSTM-M	53.57 (1.23)	60.72 (1.10)	62.52 (1.54)	64.06 (1.45)	66.43 (1.74)
	Tree-LSTM-HM	64.50 (1.53)	70.30 (1.21)	73.82 (1.24)	75.71 (1.42)	76.84 (1.44)
	BERT-H	42.41 (2.18)	58.09 (7.47)	60.15 (3.41)	62.14 (2.29)	63.57 (2.45)
	BERT-M	55.01 (5.92)	58.33 (4.07)	64.24 (0.65)	64.72 (1.64)	67.28 (1.97)
	BERT-HM	62.60 (3.25)	71.59 (1.34)	75.88 (0.79)	77.19 (0.91)	78.56 (0.36)
Gold	LSTM-H	29.14 (2.59)	15.27 (4.10)	14.48 (4.58)	10.81 (3.07)	9.65 (1.48)
	LSTM-M	78.38 (2.55)	71.32 (2.55)	63.57 (3.09)	64.96 (3.00)	67.66 (4.79)
	LSTM-HM	31.23 (6.44)	19.07 (3.25)	21.81 (4.85)	19.77 (1.33)	16.94 (3.01)
	Tree-LSTM-H	36.24 (6.63)	37.68 (11.12)	36.61 (5.18)	36.98 (0.95)	32.34 (3.50)
	Tree-LSTM-M	73.88 (2.56)	66.68 (1.42)	63.43 (3.10)	61.67 (2.45)	57.45 (3.34)
	Tree-LSTM-HM	48.45 (1.67)	37.08 (1.58)	33.32 (1.19)	28.72 (2.06)	27.33 (2.12)
	BERT-H	65.90 (3.95)	38.79 (11.03)	37.68 (10.55)	37.31 (5.10)	38.61 (5.69)
	BERT-M	74.85 (7.73)	71.28 (6.61)	64.83 (1.87)	62.37 (3.70)	59.12 (2.46)
	BERT-HM	59.54 (4.05)	42.32 (3.82)	34.85 (2.54)	29.79 (2.32)	25.99 (1.18)
Human	LSTM-H	69.62 (2.31)	84.09 (3.31)	85.65 (4.11)	88.74 (2.33)	90.85 (2.16)
	LSTM-M	23.07 (2.28)	28.87 (2.54)	35.20 (2.53)	35.09 (3.35)	32.32 (4.26)
	LSTM-HM	69.71 (2.72)	80.90 (4.10)	77.73 (5.56)	80.73 (2.12)	83.03 (2.87)
	Tree-LSTM-H	63.37 (6.04)	63.59 (8.51)	65.64 (2.96)	65.95 (0.75)	73.06 (1.97)
	Tree-LSTM-M	30.36 (1.46)	37.42 (0.90)	40.50 (2.78)	43.55 (2.50)	49.45 (3.30)
	Tree-LSTM-HM	58.36 (2.02)	67.87 (1.97)	73.97 (1.69)	78.03 (2.25)	80.11 (2.69)
	BERT-H	45.12 (3.92)	70.57 (10.04)	72.47 (6.51)	73.73 (3.43)	75.00 (4.90)
	BERT-M	28.65 (8.48)	31.27 (6.19)	39.72 (1.50)	41.90 (3.30)	46.29 (4.28)
	BERT-HM	51.33 (4.64)	68.02 (2.34)	76.19 (1.80)	79.98 (2.64)	83.09 (0.46)
Machine	LSTM-H	51.17 (2.93)	71.58 (5.59)	76.72 (6.23)	81.40 (4.00)	84.42 (2.46)
	LSTM-M	34.59 (3.02)	40.32 (2.07)	47.29 (2.79)	48.43 (4.08)	44.99 (5.66)
	LSTM-HM	66.72 (2.99)	78.38 (2.39)	78.37 (3.59)	82.60 (1.71)	83.76 (3.13)
	Tree-LSTM-H	48.79 (8.37)	48.82 (9.51)	50.72 (3.26)	55.14 (0.93)	67.48 (3.51)
	Tree-LSTM-M	65.28 (2.22)	76.98 (1.62)	79.35 (1.66)	80.74 (1.73)	82.42 (2.25)
	Tree-LSTM-HM	74.27 (2.11)	82.55 (1.67)	86.32 (1.63)	88.55 (1.69)	89.73 (1.59)
	BERT-H	32.99 (2.97)	54.41 (11.31)	57.59 (5.14)	60.87 (3.38)	62.47 (3.63)
	BERT-M	69.31 (8.18)	75.33 (5.60)	83.12 (0.71)	83.20 (2.05)	86.15 (1.48)
	BERT-HM	72.31 (4.45)	83.48 (2.15)	88.43 (1.32)	89.79 (0.63)	91.42 (0.72)

Table 6: The classification accuracies on PETCI. We report mean accuracy over 5 runs (standard deviation in parentheses). The training sets are labelled by their relative sizes, with 1 being the smallest training set, and 5 being the full training set. The suffix of model indicates the type of training set on which the model is trained.



一触即发

Yīchùjífā

4 / 5,000

Triggered

Translations of 一触即发

Adverb

Frequency

on the verge	一触即发	

Phrase

could happen at any moment	一触即发	

Send feedback

一板三眼

A board with three eyes

Alternatives:

One board, three eyes

One board with three eyes

One board and three eyes

一板三眼 ()

lit. one strong beat and three weak beats in a measure of music (four beats in the bar) (idiom); fig. scrupulous attention detail

一板一眼 ()

lit. one strong beat and one weak beats in a measure of music (two beats in the bar) (idiom); fig. follow a prescribed pattern the letter

scrupulous attention detail

Figure 2: The web interface of Google (top) and DeepL (bottom) translation. We take screenshots and perform OCR on the region in the blue box. There are other useful information in the red box, but these content are not consistently provided for each idiom, so we ignore them.