Finding ARG1 of Partitive Nouns in NomBank with DistilBERT

Ziyang Zeng
Dept. of Computer Science
New York University
251 Mercer Street, New York, NY
zz2960@nyu.edu

Abstract

This paper presents our attempts of multiple pipelines to find ARG1 of partitive nouns in NomBank with DistilBERT. NomBank is an annotation project at New York University based on Penn Treebank II corpus. Finding ARG1 of partitive nouns in the databank is an semantic role labeling (SRL) problem, which here we treated as a token classification task or a question-answering task. We adopted a knowledge-distilled version of BERT, DistilBERT, into our SRL pipeline, leveraging the knowledge of English Wikipedia and Toronto Book Corpus that the model was originally trained on. For the % dataset, we achieved an F1 score of 0.9267 on the test set. And for the partitive dataset, we achieved an F1 score of 0.7945 on the test set.

1 Introduction

NomBank (Meyers et al., 2004c) is a databank project at New York University that adds annotation layer of argument structure for instances in the Penn Treebank II corpus. Apart from the more commonly researched nominalizations of verbs and nominalizations of adjectives, it also covers relational nouns, partitive nouns and several other types of argument-taking nouns.

More recently, the NLP community has seen huge popularity in heavily adopting pretraining-based language models in attempt to transfer knowledge learned from large corpus into more specific tasks. BERT (Devlin et al., 2018) is a large-size language representation model trained on a large corpus of English text that can achieve state-of-the-art results on a variety of natural language processing tasks. BERT came a decade later than the release of NomBank and there's been little previous work using it to conduct NomBank-based tasks and related experiments. However, as powerful as BERT is, its large parameters size makes it significantly more computationally expensive to train and use. On the other hand, DistilBERT is a distilled student model of BERT that retains 97% of BERT's language understanding capabilities while being 40% smaller and 60% faster. This makes it a great candidate to train when facing with limited resources.

This paper presents multiple token classification approaches using pre-trained DistilBERT to find ARG1 of partitive nouns in NomBank. It focuses on partitive nouns (nouns that are used to describe a part or quantity of something) and the partitive task. The task is described as finding one argument (ARG1) of % (the percent sign), or in other words, finding the none group that is being sub-divided or quantified over. For example, for the sentence "Output in the energy sector rose 3.8%.", the ARG1 to be found is "Output" since it is the partitive noun that "3.8%" is referring to. This task can be translated into a binary classification problem on each token in the sentence where the model predicts whether the token is an ARG1 or not.

In addition to the token classification approaches above, this paper also experiments with question-answer task, reframing the original ARG1 finding task into the task of make the model answer the question "What is the ARG1 of this sentence?". The advantage of this adaptation is that the model would stably output exactly one ARG1 for each sentence whereas token classification could give each sentence 0 to N ARG1s.

2 Related Work

NomBank (Meyers et al., 2004c), as a databank extending the frames of NOMLEX and PropBank, annotates argument structures of common nouns similar to how PropBank annotates predicating verbs. The development of the NomBank corpus has made several other work on argument structure extraction more accessible. Jiang and Ng (2006) attempts the first NomBank-based automatic semantic role labeling system after the databank's release. Ping (2006) adapts a PropBank-based SRL system to

the SRL task of NomBank, achieving an overall F1 score of 72.73 on section 23 of the NomBank corpus.

BERT, or Bidirectional Encoder Representations from Transformers (Devlin et al., 2018), has shown state-of-the-art results on many NLP tasks. The BERT base model has 110 million trainable parameters and was trained on a concentration of English Wikipedia and Toronto Book Corpus. There have been many attempts to compress BERT into a smaller model, among which DistilBERT (Sanh et al., 2019) successfully reduces the size of a BERT model by 40%, while retaining 97% of its language understanding capabilities and being 60% faster, leveraging knowledge distillation (Bucila et al., 2006, Hinton et al., 2015) during the pre-training phase.

More recent development on semantic role labeling (SRL) has shifted from feature engineering to architecture-based modeling leveraging deep neural networks (Collobert et al., 2011). Several recent notable approaches suggest only using raw tokenized text as input and let the model learn the features from the text itself (He et al., 2017; Sahin and Steedman, 2018; Marcheggiani et al., 2017; Strubell et al., 2018). This end-to-end approach allows the model to easily generalize from one SRL task to another without very task-specific feature engineering effort.

3 ARG1 Finding Pipelines

We developed several pipelines for finding ARG1 of partitive nouns in NomBank. One pipeline utilizes maximum entropy classifiers, the others utilizes the pre-trained DistilBERT model.

3.1 Maximum Entropy Baseline Pipeline

The baseline we use is a maximum entropy machine learning method in a token-by-token classification manner. Maximum entropy is a general technique for estimating probability distributions from data. Labeled training data is used to derive a set of constraints for the model that characterize the class-specific expectations for the distribution. To perform well, the model relies on the good representation of the features we feed it. In this paper, we select features including the word stem, the neighboring 2 words, POS tags, NG-BIO tags, whether capitalized and the position of the word in the sentence. Figure 1 shows the features we selected for the maximum entropy baseline.

```
1  The STEM=the    POS=DT  BIOTAG=B-NP POSITION=0.0    NEXT_POS=NN
   NEXT_BIOTAG=I-NP    NEXT_WORD=consensus NEXT_STEM=consensus NEXT_2_POS=NN
   NEXT_2_BIOTAG=I-NP  NEXT_2_WORD=view    NEXT_2_STEM=view    NEXT_3_POS=VBZ
   NEXT_3_BIOTAG=B-VP  NEXT_3_WORD=expects NEXT_3_STEM=expect  CAPITALIZED=True
2  consensus   STEM=consensus  POS=NN  BIOTAG=I-NP POSITION=0.
   05263157894736842    PREVIOUS_TAG=@@ PREVIOUS_POS=DT PREVIOUS_BIOTAG=B-NP
   PREVIOUS_WORD=The   PREVIOUS_STEM=the   NEXT_POS=NN NEXT_BIOTAG=I-NP
   NEXT_WORD=view  NEXT_STEM=view  NEXT_2_POS=VBZ  NEXT_2_BIOTAG=B-VP
   NEXT_2_WORD=expects NEXT_2_STEM=expect  NEXT_3_POS=DT   NEXT_3_BIOTAG=B-NP
   NEXT_3_WORD=a   NEXT_3_STEM=a   CAPITALIZED=False
3  view    STEM=view   POS=NN  BIOTAG=I-NP POSITION=0.10526315789473684
   PREVIOUS_TAG=@@ PREVIOUS_POS=NN PREVIOUS_BIOTAG=I-NP
   PREVIOUS_WORD=consensus PREVIOUS_STEM=consensus PREVIOUS_2_POS=DT
   PREVIOUS_2_BIOTAG=B-NP  PREVIOUS_2_WORD=The PREVIOUS_2_STEM=the
   NEXT_POS=VBZ    NEXT_BIOTAG=B-VP    NEXT_WORD=expects   NEXT_STEM=expect
   NEXT_2_POS=DT   NEXT_2_BIOTAG=B-NP  NEXT_2_WORD=a   NEXT_2_STEM=a
   NEXT_3_POS=CD   NEXT_3_BIOTAG=I-NP  NEXT_3_WORD=0.4 NEXT_3_STEM=0.4
   CAPITALIZED=False
4  expects STEM=expect POS=VBZ BIOTAG=B-VP POSITION=0.15789473684210525
   PREVIOUS_TAG=@@ PREVIOUS_POS=NN PREVIOUS_BIOTAG=I-NP    PREVIOUS_WORD=view
   PREVIOUS_STEM=view  PREVIOUS_2_POS=NN   PREVIOUS_2_BIOTAG=I-NP
   PREVIOUS_2_WORD=consensus   PREVIOUS_2_STEM=consensus   PREVIOUS_3_POS=DT
   PREVIOUS_3_BIOTAG=B-NP  PREVIOUS_3_WORD=The PREVIOUS_3_STEM=the NEXT_POS=DT
   NEXT_BIOTAG=B-NP    NEXT_WORD=a NEXT_STEM=a NEXT_2_POS=CD
   NEXT_2_BIOTAG=I-NP  NEXT_2_WORD=0.4 NEXT_2_STEM=0.4 NEXT_3_POS=NN
   NEXT_3_BIOTAG=I-NP  NEXT_3_WORD=%   NEXT_3_STEM=%   CAPITALIZED=False
```

Figure 1: Sample sentence features for maxent system

3.2 Plain Raw Input Pipeline

The simplest pipeline for finding ARG1 of partitive nouns in NomBank assumes that the input text is raw and unprocessed. The pipeline first tokenizes the text using the tokenizer provided by DistilBERT. Then, the model predicts whether each token is an ARG1 or not. The pipeline is illustrated in the following figure 2.

However, the NomBank dataset provisions text as already tokenized. The way the tokenizer provided by DistilBERT is often more fine-grained than how the NomBank dataset provides the text, causing some aligning issues. We fix this by align the tokens using `word_ids` in the tokenization output. This is explained in detailed in section 4.2.

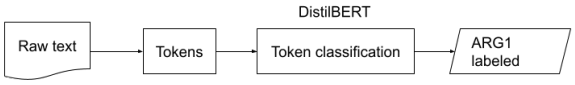

Figure 2: The simplest pipeline for finding ARG1 of partitive nouns in NomBank

In practice, this pipeline has the tendency of over generating ARG1 labels. It predicts on a per-token basis, making classification of whether the token is an ARG1 or not on every token, thus possibly producing 0 to multiple ARG1s in one sentence. It is shown by our experiments that the deep models are more common to produce more than 1 ARG1s in one sentence. The ground truth, however, is one and only one ARG1 per sentence. A treatment of this inconsistency is to always force one ARG1 by the logits output of the deep neural network. We assign the token with the largest logit value in the output vector for the whole sentence to be the ARG1.

3.3 Feature Enhanced Pipeline

NomBank provides a number of features for each token in the databank, including POS tags and NG-BIO tags. They could be helpful for the task of finding ARG1 of partitive nouns. The pipeline below incorporate these features to the raw text input in 3.2 to find ARG1 of partitive nouns. The pipeline is illustrated in the figure 3.

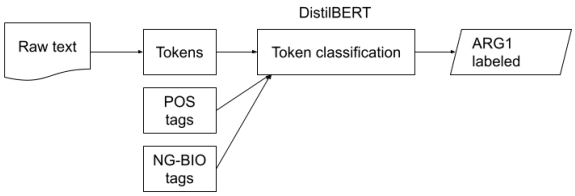

Figure 3: The feature-enhanced pipeline for finding ARG1 of partitive nouns in NomBank

Because the original pre-trained DistilBERT model does not take extra features including POS and NG-BIO tags as input, we have to modified the token classification model and alter the last linear layer, concatenating the original BERT hidden layer output with the extra features, then doing token classification through the linear layer.

3.4 Question-Answer Pipeline

Apart from token classification, we also explored question-answer approach which rephrase the original problem into a QA problem. Question-answering, just as the plain token classification approach, assumes tokenized input and no extra features. Additionally, question-answering tasks also requires a question prompt and answer location for each sample while training. The pipeline is illustrated in the figure 4.

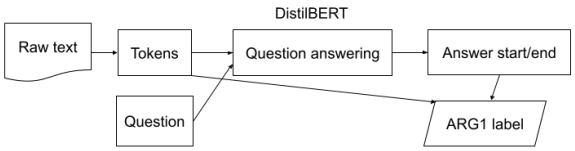

Figure 4: The question-answering pipeline for finding ARG1 of partitive nouns in NomBank

The output of the question-answering model is the location (start index and end index) of the predicted answer. By locating the token on the original input token list with the start index and end index, we can reconstruct the ARG1 label of the sentence. The answer relocation is a bit more complicated due to the tokenization discrepancies between the original input and the tokenized input. Again, this is explained in detailed in section 4.2.

4 Experiments

4.1 Dataset

The dataset we use in this task is part of the NomBank that specifically focus on partitive nouns. In the dataset is split into training, dev and test set. The training set has 2367 sentences or 66186 words, the dev set has 83 sentences or 2225 words, and the test set has 150 sentences or 4276 words. Each token in a sentence takes a line in the data file, the line containing the token itself, the POS tag, the NG-BIO tag, the index of the sentence and the index of the token in the sentence. The semantic role label is put at the end of each line, if the label is not one of "ARG1", "PRED" and "SUPPORT", then it's omitted. Figure 5 is one sample sentence from the training dataset.

```
1   But CC  O   0   0
2   about   IN  B-NP    1   0
3   25  CD  I-NP    2   0
4   %   NN  I-NP    3   0   PRED
5   of  IN  B-PP    4   0
6   the DT  B-NP    5   0
7   insiders    NNS I-NP    6   0   ARG1
8   COMMA   COMMA   O   7   0
9   according   VBG B-PP    8   0
10  to  TO  B-PP    9   0
11  SEC NNP B-NP    10  0
12  figures NNS I-NP    11  0
13  COMMA   COMMA   O   12  0
14  file    VBP B-VP    13  0
15  their   PRP$    B-NP    14  0
16  reports NNS I-NP    15  0
17  late    RB  B-ADVP  16  0
18  .   .   O   17  0
```

Figure 5: One sample sentence in the training dataset

In a sentence in the % dataset, the PRED labels % symbol, the ARG1 is the partitive noun that % symbol refers to, and the SUPPORT marks the word connecting the ARG1 and the PRED. It is guaranteed that there is one and only one ARG1 in each sentence of the % dataset.

4.2 Data Preprocessing

Even though for deep neural network input wouldn't need much feature engineering, we still need to tokenize the text and convert the text into numerical representation so that it can be served

as the input of the model. For DistilBERT-based pipelines, the tokenizer provided by DistilBERT is used to tokenize the text. However, the dataset is already tokenized and the way DistilBERT tokenizes a sentence could be different from how the sentences are already tokenized in the dataset. More specifically, the tokenizer from DistilBERT could tokenize the sentence into even smaller tokens than the tokenizer used in the dataset. Therefore, this could cause the output label of the model fail to align with the input given. Fortunately, the results of the DistilBERT tokenizer include the align information for the tokenization. With this information, we can reconstruct the tokenized sentence from the DistilBERT tokenizer output and match the model output with the original input.


```
{
    "input_ids": [101, 1003, 1997, 2054, 1029, 102, 2021, 2055,
    2423, 1003, 1997, 1996, 25297, 2015, 1010, 2429, 2000, 10819,
    4481, 1010, 5371, 2037, 4311, 2397, 1012, 102, 0, 0, 0, 0, 0,
    0, 0, 0, 0],
    "attention_mask": [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
    1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0],
    "start_positions": 12,
    "end_positions": 13
}
```


Figure 6: One sample sentence for QA task

A special treatment for the question-answering pipeline is that the ARG1 label has to be transformed into the answer location (the start and end indexes). And the question prompt for every sample is the same "What is the ARG1 of the sentence?". The exact same question prompt is used for validation and testing. Figure 6 is one processed sample sentence for the QA task, inputs are padded in batch and stacked as matrix to be able to be processed by the model in batches. `input_ids` are the numerized tokens of both the question prompt and the input text concatenated by `102`. `attention_mask` is the attention the model should be given to the tokens, `0` means the token should not be considered by the model. `start_position` and `end_position` are the start and end indexes of the answer in the input text.

4.3 Experiment Settings

All deep models used in this paper are implemented in PyTorch and with HuggingFace transformers libraries. The pre-trained models are fine-tuned on a single NVIDIA P100 (16GB) GPU on the Google Colab platform. The baseline maximum entropy classifier implementation we used is `opennlp.maxent` by Baldridge et al. The deep models are fine-tuned for 10 epochs with the learning rate of `2e-5` and weight decay of `0.01`. For token classification tasks, the batch size is 64. For question-answering tasks, the input would be bigger with the question prompt, so the batch size is set to 32 to be able to fit into the GPU's RAM.

4.4 Evaluation Metrics

To evaluate the performance of the model, we used precision, recall and F1-score as evaluation metrics. The precision and recall are calculated by the number of correct predictions divided by the total number of predictions. The F1-score is calculated by the harmonic mean of precision and recall. Below are the formulas for calculating the precision, recall and F1-score, in which TP is the number of true positives, FP is the number of false positives, FP is the number of false positives and FN is the number of false negatives.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3}$$

4.5 Partitive Dataset

Apart from the original % task, we also experimented the token classification of DistilBERT on the combined dataset of % and partitive datasets of NomBank. Different from the % dataset, there's not only ARG1 but also ARG0, ARG2... and more role labels in the partitive dataset. In order to adopt the same pipelines, we preprocessed the partitive dataset into the same format as the % dataset and combine them together into a larger dataset. The larger dataset is also split into training, dev and test set. The training set has 12991 sentences or 380247 words, the dev set has 426 sentences or 12671 words, and the test set has 746 sentences or 21536 words.

The partitive dataset, in comparison with the % dataset, has more distractions in terms of ARG1 finding task since ARG0, ARG2 and other labels are given. Also, there could be more than one or zero ARG1s in one sentence, thus making our QA and ONE ARG1 pipelines useless since the tendency of producing only one ARG1 is no longer preferred.

Model	Precision	Recall	F1-Score
MaxEnt Baseline	71.88%	61.33%	66.19%
DistilBERT	93.75%	90.00%	91.84%
DistilBERT (POS+BIO)	93.19%	91.33%	92.25%
DistilBERT (QA)	92.00%	92.00%	92.00%
DistilBERT (ONE ARG1)	92.67%	92.67%	92.67%

Table 1: Precision, Recall and F1-Score on the % dataset

Model	Precision	Recall	F1-Score
MaxEnt Baseline	55.33%	36.02%	43.64%
DistilBERT	80.49%	78.43%	79.45%
DistilBERT (POS+BIO)	81.53%	76.94%	79.16%

Table 2: Precision, Recall and F1-Score on the partitive dataset

5 Experiment Results

We experimented with multiple ARG1 finding models and pipelines on both % dataset and partitive dataset from the NomBank corpus. The results are shown in Table 1 and Table 2 respectively.

For the partitive dataset, because there could be zero or multiple ARG1s in one sentence for ground truth, there's no need to force model to predict only one ARG1, thus we abandoned the QA and ONE ARG1 pipelines for the partitive dataset.

5.1 % Dataset

As shown by Table 1, the MaxEnt baseline achieved an F1 score of `0.6619`. The DistilBERT models perform much better than the MaxEnt Baseline model on the % dataset generally even when given less features, achieving an F1 score of 0.9184 for the basic pipeline, `0.9225` for the feature-enhanced pipeline, `0.92` for the QA pipeline and `0.9267` for the ONE ARG1 pipeline. However, The performance differences between different DistilBERT-based pipelines are very minor and can possibly be counted as experimental error. The introduction of additional POS tags and NG-BIO tags features likely don't seem to improve the model's performance in any statistically significant way.

A major advantage of DistilBERT pipelines over the baseline is the limited number of features needed. The basic and the QA pipelines only require the text input itself and feature-enhanced one requires the text input, the POS tags and the NG-BIO tags.

By a closer look to where the model struggles to produce correct predictions, Figure 7 being one of the examples, we can see that the model is trying to predict more than one ARG1s in the sentence. In the given example, the ground truth is only `tons` and the model is trying to predict `tons` and also `steel`. Though being over generating ARG1s, this is actually a very sensible prediction since "tons of steel" is indeed a noun group that the % symbol is referring to. And our solutions to this over generating problem (QA task and forcing one ARG1) didn't quite improve the performance. In our experiment, we found that, when faced with multiple ARG1 candidates in a noun group, the model tends to choose the head noun as the ARG1, which in this case is `steel` instead of the ground truth `tons`.

```
739  of
740  the
741  100
742  million
743  tons    ARG1
744  of
745  steel   ARG1
746  used
747  annually
748  by
749  the
750  nation
751  .
```

Figure 7: One over-generating prediction sample for DistilBERT

5.2 Partitive Dataset

As for the partitive dataset, it is also shown by Table 2 that the DistilBERT model performs better than the MaxEnt Baseline model which achieved an F1 score of `0.4364`. The basic DistilBERT pipeline achieved an F1 score of `0.7945`. It is also observed that the POS tags and NG-BIO tags features

didn't have statistically significant improvement on the performance of the model, achieving an F1 score of only `0.7916`. Aside from that, when compared to the % dataset, the decrease in performance is expected since the partitive dataset has more distractions and noise. The tendency of over generating ARG1s is also observed for DistilBERT models in the partitive dataset, and can not easily be improved by forcing one ARG1, which is consistent with the observations made for the result on % dataset shown in 5.1.

6 Conclusion

This paper has discussed the semantic role labeling task of finding ARG1 of partitive nouns in NomBank. And we have experimented with multiple ARG1 finding models and pipelines on % dataset and partitive dataset from the NomBank corpus, taking advantages of the state-of-the-art DistilBERT model, then successfully developed a SRL system that can achieve an F1 score of `0.9267` on the test set of the % dataset and an F1 score of `0.7945` on combined dataset of both % dataset and partitive dataset. In comparisons with the baseline model, pre-trained DistilBERT models have shown their great advantage of text representation in this specific NLP task.

By the results of our experiments on feature-enhanced pipelines, we also show that the DistilBERT model cannot be significantly improved by adding POS tags and NG-BIO tags features. This is consistently shown by our experiments on both % dataset and partitive dataset. Besides, for the % dataset, question-answering variant and forcing one ARG1 variant of DistilBERT models have not shown much significant improvement on the performance of the model either. However, the QA approach is indeed a novel perspective to look at the task of finding ARG1, reframing a SRL task into a QA task which is a more open and flexible task. The flexibility comes with the free choice of prompt question - a prompt question can be any text. Therefore, what could possibly be further investigated is the question prompt we use to ask the model to find ARG1.

Based on the performance decrease observed in the partitive dataset, we also concluded that the ARG1 finding pipelines are generally sensitive to the noise in the partitive dataset. However, the DistilBERT models are relatively more robust to noise with 14.26% decrease in performance compared to the baseline MaxEnt model's 34.06% decrease.

7 Discussion and Future Work

The two extra POS tags and NG-BIO tags features didn't seem to improve the performance of the model in a statistically significant way. This could be due to the huge amount of knowledge the deep model has already gained from the pre-training stage, thus rendering the addition of these two features insignificant. However, feature engineering in general could possibly be helpful in improving the model. We could investigate more selections of features and try to incorporate them into the pipelines we have developed. The features that are not covered by the pre-trained DistilBERT models could help the model in the task of finding ARG1.

Question-answering pipeline is a novel approach to the task of finding ARG1 of partitive nouns in NomBank. As mentioned, what we haven't explored enough on is the question prompt we use to ask the model to find ARG1. The question prompt can be any text, but the way we framed the question could be the key to how well the model would comprehend the task and execute it. For future work, we'd like to explore more kinds of question prompts and analyze the effect of different question prompts on the performance of the model.

The system we developed has shown good performance on the % dataset. However, in real-world applications, like shown on the partitive dataset, the noise could majorly impact the performance of the model. The directions of reducing the impact of noise could be exploring preprocessing techniques in attempt to remove the noise, or architectural changes in the model to make it more robust to noise. A plausible idea is to use BERT output as an embedding of the token, then feed it with other feature to an RNN model to predict the ARG1. More trainable layers for the token classification task could make the model more robust to noise.

References