Detecting and Resolving Referential Ambiguity

Merve Gül Kantarcı[a], Güray Baydur[b], Çağlar Fırat[c] and Baturalp Yörük[d]

[a]*Boğaziçi University, Istanbul, Turkey*
[b]*Boğaziçi University, Istanbul, Turkey*
[c]*Boğaziçi University, Istanbul, Turkey*
[d]*Boğaziçi University, Istanbul, Turkey*

Abstract

Sentences in a natural language may include different kinds of ambiguities. Here, our focus is on referential ambiguity on pronouns. We try to detect whether there is referential ambiguity or not. If there is referential ambiguity, then we try to resolve it by finding the corresponding antecedent. Our methodology is based on GPT-2 language models.

Keywords

Referential Ambiguity, ReqEval, GPT-2

1. Introduction

When a sentence can be interpreted in more than one way, it is called to have an ambiguity in that sentence. If there is an ambiguity, since it can be interpreted to different meanings, it may easily cause some problems. When it comes to natural language processing tasks or automated requirement elicitation tasks, detecting and resolving ambiguities are becoming more and more important than it is in real life.

Manually detection of ambiguity is a time consuming task when we think of huge data samples. It may also cause some errors since people who are detecting it can make mistakes normally.

When we have considered the reasons above, we decided to create a tool for detecting the ambiguities in the sentences. Our tool takes a input file which includes one or more sentences and gives an output which implies for every sentence whether it has referential ambiguity or not.

Also, after detecting whether a sentence has referential ambiguity or not we want to resolve that ambiguity and try to give as an output which antecedent is referred by the pronoun. Furthermore, we are also trying to resolve more than one referential ambiguity if there is.

We have tried different methods while trying to realize the above mentioned tasks. However, we have failed from some of the methods and decided to go with some other. We are going to

merve.kantarci@boun.edu.tr (M.G. Kantarcı); guray.baydur@boun.edu.tr (G. Baydur); caglar.firat@boun.edu.tr (Ç. Fırat); baturalp.yoruk@boun.edu.tr (B. Yörük)
https://github.com/guraybaydur/ (G. Baydur); https://github.com/c-firat (Ç. Fırat); https://github.com/baturalp-yoruk (B. Yörük)
0000-0002-0877-7063 (M.G. Kantarcı); 0000-0001-7116-9338 (G. Baydur); 0000-0002-9421-8566 (Ç. Fırat)

ReqEval Workshop Proceedings (nlp4re.github.io/2020/reqeval.html)

mention both our failed trials and the successful one.

2. Related Works

Before GPT-1, natural language processing systems focuses on specific tasks like textual entailment and classification of sentiment. Most of the tasks use supervised learning. There are two limitations in such type of tasks. One of them is the need for large scale data set and most of the times it is not available. The other one is that they can not be generalized. For this reason, OpenAI initiative proposed a new language model using unlabeled data for unsupervised learning [1]. They fine-tuned the model with examples form specific tasks like classification, sentiment analysis and so on.Task specific input transformations are used in GPT-1. 12-layer decoder is used by GPT-1 with only transformer structure combined with masked self-attention to train language model.

GPT-2 introduces improvements to GPT-1 model, by using a large scale data set and increasing the number of parameters to the model to strengthen the model [2]. A concept called zero shot task transfer and zero shot learning is applied with GPT-2. Zero shot learning is a special case where no examples are provided at all and instructions are given and model understand according to that.

Task conditioning is also preferred in GPT-2, where, for different tasks, the model is trained to produce different outputs for same input. The basis for zero-shot task transfer is formed with task conditioning which is mentioned recently.

To sum up the paper, it is stated that the performance increases log-linearly as the capacity increases. However, when WebText dataset is used, GPT-2 under fitted. This indicates that larger language models need to be constructed. This led to new paper about GPT-3 [3], however due to lack of access to GPT-3 model, this paper is not investigated further.

3. Method

3.1. Ambiguity Detection

For ambiguity detection, we tried to approach detecting the ambiguity problem in a sentence with some heuristics. First of all, it is known that when a single sentence starts with some pronoun and that sentence is not preceded by another sentence, the sentence can be considered as ambiguous. Hence, in case of rule based ambiguity detection, this rule can be put in the first place inside rule set.

For any sentence that does not begin with a pronoun without prior sentence, further investigations are needed to detect ambiguity. One attempt to solve this problem may include counting noun phrases inside the sentences and compare it with the count of pronouns in the sentence. This intuition came from the idea that if the sentence is simple and has only one noun chunk and one pronoun, it is trivial to mark the sentence as unambiguous. Furthermore, if the sentence contains multiple noun phrases and multiple pronouns, it may be more likely to contain ambiguity with respect to number of components to correlate. Therefore, for each sentence in the data set, we calculated pronoun count and noun phrase count. By dividing the pronoun count to noun phrase count, we constructed a ratio. Then we sum up all the corresponding ratios for the ambiguous sentences as well as unambiguous sentences. Then we compared the results. The computation showed that the sums are really close to each other (ratio of ambiguous: 0.23, ratio of unambiguous: 0.25). As a result, we can not tell anything clear with this rule. Hence it is omitted.

Another approach to solve the problem consists of detecting ambiguity with the help of conjunctions. To elaborate more on this, a sentence with a conjunction may include a pronoun that might create ambiguity in overall context. For this idea, we tried to detect whether a sentence contains a conjunction (”and”, ”or”) or not and increment the count if it does. We did the same for both type of texts. Finally, the results demonstrate that the counts of conjunctions in ambiguous and unambiguous texts are similar to each other (or and counts ambiguous: 28 , or and counts unambiguous: 29) and therefore unnecessary to add the rule to the rule set.

3.2. Ambiguity Resolution

3.2.1. Binary Classification for Detecting Correct Antecedent

Two attempts made for ambiguity resolution. The first attempt includes training a binary classifier to detect whether a candidate antecedent is the right candidate for the given pronoun or not. In other words, given two words in which one of them refers to candidate antecedent and the other one refers to the pronoun to be resolved, check if it is the right candidate or not. For that purpose, we constructed a feature set with some heuristics. Feature one can be defined as plurality of the two given words. This feature checks whether two words (pronoun and candidate antecedent) are both plural or not. The reason behind this feature lies on the fact that pronouns like ”they”, ”their”, ”them”, ”theirs” may refer to plural noun or noun phrases. Therefore if we try to find what ”they” refer to in the given sentence, we only need to consider plural nouns for resolution and this property will give us a clue for that. Second feature in our feature set is syntactic dependency of two words because for the ground truth, each word may have the same syntactic responsibility in the sentence. We derived this feature using Spacy tokenizer. Third feature is closeness of two words, in other words, the absolute difference of start indices of two words. We computed this feature since if real candidate and pronoun may lie close to each other in the sentence. Fourth feature is detecting both words have same gender representation. To exemplify, if a masculine noun is detected, its proper resolution will be ”he”. Same applies for feminine and neutral nouns. Final feature is detecting if following word of the two words are verbs or not. This feature is deduced from the idea that candidate and pronoun may follow the same verbs in the sentence.

The first three features (plurality, syntactic dependency, closeness) is constructed properly for each sentence. However other two features can not be constructed due to lack of time. It will be considered as a future work. The source code can be reached from here

3.2.2. Training A Language Model

GPT-2 [2] is known to be a successful language model on text generation. Considering that referential ambiguity resolving requires text generation at some point, we want to try out its phenomenal performance on the task. Main challenge to pursue this method is the limited size of the dataset. Since the task is specific, and language model is very large, we expect that it would require a large chunk of data. One point particularly takes attention regarding to limitation of the dataset that the disambiguation task consists of only two requirements with two unambiguous referential problem. It is a big challenge for the modelwith the current setting to differentiate between one solution and two solution with seeing only two examples of it.

Preprocessing First step is to ensure that input sequence is well formatted for GPT-2 [2] model. A single input sequence is defined between "<|startoftext|>" and "<|endoftext|>" tokens. When there are unambiguous multiple references in a given requirement solutions are separated with a newly introduced token "<next>". In order to standardize the notation all the occurrences of "<referential id=<id»" is converted to "<referential>". In the early experiments to solve this task, we missed a key point in formatting by underestimating the effect of whitespace. In original dataset file <referential> and </referential> tags are adjacent to pronoun that is to be predicted. This is against the tokenization rule and its hard to distinguish them when it is not separated by whitespace.
Below example shows a single input sequence that is fed to GPT-2 model:
<|startoftext|> The devices can be manually controlled/operated from <referential> their </referential> cabinets (e.g.": Gates). the devices <|endoftext|>
An example with 2 references:
<|startoftext|> Each rate group has some number of tasks associated with <referential> it </referential> and <referential> it </referential> also has a rate for those tasks. each rate group <next> each rate group <|endoftext|>

Experiments GPT-2 model comes in different sizes but considering that the smallest one includes approximately 124 million parameters, it is hard to meet the hardware expectations to run the all versions of it. Therefore we experimented with the model with 124M parameters and 355M model parameters. They are referred as small and medium for the rest of this paper. Both small and medium size models are trained 100, 1000 and 2000 steps to evaluate the performance. Both small and medium size models are trained using 100% of provided dataset and 80% of provided dataset. When the training set is reduced to %80 of it, the rest is used as test dataset to evaluate how much, or if, model memorizes and overfits the dataset. Best mode chosen is as medium size with 2000 steps training. Qualitative results and quantitative results of the models regarding to test and train accuracy are discussed in Evaluation The source code can be reached from here

4. Evaluation

4.1. Ambiguity Detection Mechanism

To detect ambiguity, some ratios are found and examined from the data set. These ratios are listed below.

$$R_1 = \frac{\#pronoun}{\#nounchunks}$$

$$R_2 = \frac{\#or\_and\_cnt\_of\_ambigous\_sentence}{\#or\_and\_cnt\_of\_unambigous\_sentence}$$

$$R_3 = \frac{\#potentialAntecedentsofAmbigousSentence}{\#potentialAntecedentsofUnmbigousSentence}$$

$$R_4 = \frac{\#pronounCountofAmbiguous}{\#pronounCountofAmbiguous}$$

For the data set, these R_1 and R_2 ratios are found as almost same for ambiguous and unambiguous sentences. Therefore, these two ratios couldn't be evaluated to use in detection mechanism, however R_3 and R_3 are found as slightly usable to use in detection mechanism, because the potential antecedent number for ambiguous sentences are 1.2 times higher than unambiguous sentences and the pronoun counts are also almost 1.2 times higher in ambiguous sentences.

4.2. Ambiguity Resolution Mechanism

4.2.1. Quantitative Evaluation

Even though our later evaluation shows that model overfits, we find the medium size model with 2000 steps most consistent and successful. Precision and recall metrics with perfect and partial match are presented in Table 1.

4.2.2. Qualitative Evaluation

Even though the test performance was not promising, we go out of scope of requirements and experiment with more casual sentences. Most interesting examples are presented in Table 2. This evaluation metric shows that the model learns well the grammatical relation but not strong in semantic relation. The evaluation is made with medium size model with 2000 steps training on whole dataset provided.

5. Future Work

As a future work, our failed methodologies can be implemented successfully. They can be improved further and may give even better results from our successful method.Also, our successful method's success rates (i.e. recall, precision) can be improved by changing the sample size or training method or many other variables. The more trials on the combination of some variables, the more chance to find the optimal variation.

Model	Precision (Perfect)	Precision (Partial)	Recall (Perfect)	Recall (Partial)
Small	0.77	0.79	0.75	0.77
Medium	0.96	0.99	0.94	0.97

Table 1. Quantitative Results Precision and recall are calculated using results of medium size model with 2000 steps training.

Input	Output
The road was not comfortable for driver, and he was looking frightened.	the driver
The road was not comfortable for driver, and it was looking dangerous.	the road
The black cat was sitting in a room, and it was very crowded.	the black cat
The black cat was sitting in a room, and it was staying still.	the black cat
The little children were around abandoned house, hence they were looking upset.	the children

Table 2. Qualitative Results Row 1 and 2 show that model grammatically learns basics of resolving a reference. We doubted that if that was an indication of semantic differentiation between references. However, row 3 and 4 proven that semantic realise is not very low despite that we might expect from a successful model for this task. One other observation is model predicts the input 3 as ”the black cat”, which includes adjacent adjective. However, the last row does not reflect the same behaviour. This might be caused by inconsistency in train data, overfitting or lack of learning.

Considering that the limitation of the dataset size was problematic with the methodology we pursue, it is clear to investigate further perspectives on this topic. First approach seems to be possible and feasible in this context: data augmentation and recent techniques that research on learning with small dataset. The first approach is very promising on this task since this type of learning that task requires may benefit a lot from word replacement techniques. Also very recently, a framework [4] that heavily leans on such techniques and data augmentation is introduced that offers a feasible and exciting solution to mentioned problem in this paper.

Unreported experiments on GPT-2 to jointly solve the detection and disambiguation look promising. With few human evaluations on the issue, it is shown that when the model gets closer to learn disambiguation task, the generation becomes more inconsistent when a sentence with ambiguous reference is prompted. This is a clear implication that model might be able to distinguish between unambiguous and ambiguous references. This shows that with a different approach the model might be a solution to the both tasks.

6. Conclusion

Ambiguity causes some understanding problems in sentences. It is time consuming and open to errors to solve ambiguities manually. We have worked on referential ambiguity detection and ambiguity resolution.

Firstly, we tried to detect whether there is a referential ambiguity or not. Secondly, after finding a ambiguity, we tried to resolve this ambiguity and find the corresponding antecedents in the sentences.

Of course our experiment results could be better, but we have tried some different methodologies and tried to find the best methodology to detect and resolve referential ambiguities. Here we propose some points to future investigators of the topic.

References

[1] A. Radford, Improving language understanding by generative pre-training, 2018.

[2] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners, 2019.

[3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krüger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, D. Amodei, Language models are few-shot learners, ArXiv abs/2005.14165 (2020).

[4] J. X. Morris, E. Lifland, J. Y. Yoo, J. Grigsby, D. Jin, Y. Qi, Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp, 2020. arXiv:2005.05909.