Ambiguity in Humor: Detecting English Puns

Viktoriia Tsosenko
Computational Linguistics Department
Heinrich Heine University Düsseldorf
Viktoriia.Tsosenko@hhu.de

October 8, 2025

Abstract

Humor is a complex linguistic phenomenon, and puns are one of the good examples of how ambiguity creates amusement. Detecting puns automatically remains a challenging task for natural language processing systems because it requires understanding lexical ambiguity and context. This paper focuses on the automatic detection of homographic puns, which is the first step in computational humor analysis. For this purpose, a publicly available, pre-annotated dataset the *SemEval-2017 Task 7* is used, together with a pretrained BERT-based language model. The BERT model is fine-tuned on the *SemEval* dataset to classify sentences as either containing a pun or not. To achieve this, PyTorch is used for training. The goal of this work is to implement a simple but effective model for pun detection and to get experience with data handling and machine learning. The findings may provide a starting point for more advanced research, such as pun location or interpretation, and highlight the difficulties of modeling humor computationally.

1 Introduction

Human communication is difficult to imagine without humor. "If computers are ever going to communicate naturally and effectively with humans, they must be able to use humor." (Binsted, 2006, p.59). Humor also has its "beneficial aspects, because it banishes sadness and boredom, puts the individual in an optimistic mood and, in general, lightens the load of everyday living." (Larkin-Galiñanes, 2017, p.4). Making machines *understand* and *produce* humor is therefore beneficial for people. Among the various forms of humor, puns are particularly interesting because they exploit the multiple meanings of words or the similarity of sounds to generate unexpected interpretations. It is known that words often have multiple meanings, and Zipf (1949) indicates that the most frequently used words tend to have more senses than less frequent ones.

Studying puns is not only relevant for humor research, but also useful in real-life applications. Automatic pun detection can improve tasks such as chatbot interaction, sentiment analysis, and figurative language understanding, where recognizing humor or ambiguous language is essential to respond naturally. This paper focuses on homographic pun detection, which occurs when one and the same word carries multiple meanings in a sentence. The *SemEval-2017 Task 7* dataset provides English sentences labeled as a pun or not a pun. The task represents a binary classification problem: given a sentence, the model must decide whether it contains a pun. The research objectives are as follows:

1. To preprocess and prepare the *SemEval-2017 Task 7* dataset for model training
2. To fine-tune a pre-trained BERT model for the binary classification of sentences as pun or non-pun
3. To evaluate the model's performance using standard metrics such as accuracy and F1-score
4. To analyze the results of transformer-based approaches in understanding humor and ambiguity

2 Humor and Ambiguity

The perception of what humor is has undergone major changes throughout the history. Larkin-Galiñanes (2017) believes that humor as we know it today, originates from the 20th century. There is no single, universally accepted definition of humor though. Some writers believe it is even impossible to define what exactly humor, as a term, means (Attardo, 1994).

Plenty of linguistic devices are used to create humor. In Ancient Greece and Rome, Aristotle and other classic writers tried to find out what linguistic techniques were used to produce laughter, and ambiguity was listed among them. In Rhetorics, Aristotle talks about "twisting from the proper and apparent sense", or giving a word "a different 'turn'" that awakes laughter. He mentions "temporary deception practiced upon the listener", comparing it with some kind of riddle. When listening to a joke, the listener "does not at once trace the resemblance", but by "discovering its meaning, the listener learns something new" and is therefore experiencing a satisfying mental "aha!" moment. (Aristotle, Translation, 2009, p.319-320). This is a good evidence that ambiguity in humor has interested people starting from Ancient Times.

The examples below illustrate how ambiguity works at different linguistic levels (morphological, lexical, syntactic etc.), and how humor depends on it.

Structural ambiguity Dubinsky presents a joke from the Dilbert cartoon (Dubinsky, 2011, p.56):

> *Person A to Person B:* Would you like to buy advertising in my new magazine called Gullible World? We have between one and two

billion readers!

Some time later, Person A to Person C: I figured out how to make three readers sound like a lot.

The humorous phrase *one and two billion readers* could mean:

1. `[[one and two] billion]`
2. `[one] and [two billion]`

Compound words (Dubinsky, 2011, p.44):

1. The word *man*: "while a milkman delivers milk, you really don't want your garbagemen or firemen bringing garbage or fire".
2. "In noun + noun compounds, sometimes the first noun tells you what the second one is made of or with, as in *cheesecake* or *apple pie*. Sometimes you hope that it doesn't, as in *shepherd pie* (more commonly known as *shepherd's pie*) or *Girl-scout cookies*. Or you find out that it does, and wish you hadn't, as in *headcheese*."

Idioms (Dubinsky, 2011, p.46), illustrated with an idiom *a piece of me* (meaning "challenge to fight") and its literal meaning:

1. Frankenstein's monster gets into an argument at a bar and says: *"You want a piece of me? Huh? Huh? You want a piece of me?"*.

3 Classification of Puns

"Punning is a form of humorous wordplay based on semantic ambiguity between two phonologically similar words – the pun and the target – in a context where both meanings are more or less acceptable." (Palmann, 2025, p.1)

There are different types of puns. The most relevant ones for natural language processing application are homophonic and homographic puns.

Homophones Homophonic puns sound the same, but their spelling is different. Dubinsky (2011, p.51) refers to a clear example from a 2001 FoxTrot panel:

1. Jason Fox and his friend Marcus show seven carved pumpkins in a row. On the pumpkins, they carved, in order: the number "3," a decimal point, then "1," "4," "1," "5," and "9."Jason tells his sister: *We're calling it "pumpkin pi."*

The joke relies on the homophone pair *pie* (as in pumpkin pie) and *pi* (a mathematical constant).

Homographs Homographic puns share the same spelling, as in:

1. I used to be a banker, but I lost *interest*.

Interest can mean curiosity and money earned on savings.

It is important to mention congruity between the two senses when talking about puns. "Nothing is more futile than the irrelevant pun that is based on only a verbal similarity and brings out no contrast, innuendo, or congruity of meaning" (Stanford, 1972, p.72). With no semantic opposition in the punning text, it will not be a joke anymore (Hempelmann et al., 2017). Raskin expressed that puns, based "on purely phonetical and not semantical relations between words" are called "bad puns" (1985, p.116).

Understanding a pun often requires recognizing multiple meanings simultaneously, which is easy and intuitive for humans, but remains difficult for machines. Next chapters are dedicated to the automatic pun detection, namely of homographs.

4 Dataset

The *SemEval-2017 Task 7* includes six separate files with homographic puns and six with heterographic ones. For this study, only the homographic pun dataset is relevant, which consists of three pairs of files: each pair includes an `.xml` file with sentences and a corresponding `.gold` file with annotations. It is structured as follows:

Subtask 1: Pun Detection

The `.xml` file contains 2250 annotated sentences, each with its own *sentence ID*.

Each token in a sentence also has its own *ID*.

The corresponding `.gold` file provides labels for each sentence ID (1 for pun, 0 for no pun)

Subtask 2: Pun Location

The `.xml` file includes only pun sentences (1607).

The `.gold` file specifies the position of the punning word in each sentence with its ID.

Subtask 3: Sense Annotation

The `.xml` file contains 1298 sentences with puns, where the punning word is annotated with two different WordNet sense IDs, representing its two possible meanings in context.

The `.gold` file with these annotations to indicate the correct senses.

For this paper, only **Subtask 1: Pun Detection** was used, split into 3 subsets, namely: a training set (70%), a validation set (15%) and a test set (15%).