For sentence based ambiguity detection, we chose to use a probabilistic approach. In essence, we calculated the probability of each bigram being involved in an uncertain sentence, and then calculated the probability of a sentence being uncertain based on the n-grams it was composed of.

We tried a few different approaches in this direction. Our first attempt consisted of simply looking at words that start an uncertain phrase in the training data, and classify these as hedge words. We counted the number of times a word appears as a hedge word, as well as the number of times it appears in total, and computed the probability of a word being a hedge word. Then, when classifying a sentence as either uncertain or not, we looked at the probability of every word in the sentence being a hedge word. Then, we compared the highest of these probabilities to a threshold to classify the sentence.

While our initial approach worked reasonably well in most cases, we noticed a few cases where it had trouble. The first was a word having multiple forms, and being uncertain in one and not in another. Consider, for example, the word “May” – it can refer to the month (“My birthday is in May”), which carries no uncertainty, or it may refer to something that happens some of the time (“The car may be green”), which is obviously very uncertain. To tackle this, we used part of speech tagging to disambiguate different forms of words.

Another approach we tried for the sentence level uncertainty detection problem was using n-grams. Specifically, in the training data, we looked at groups of words (n words, specifically) that occurred at the start of an uncertain phrase. This was to accommodate sentences such as “Some people claim …”, which, while the sentence is uncertain as a whole, contains no words that are inherently uncertain. We tried extending the word both forward and backward, in order to capture information that occurred before a cue as well as after one.

However, based on extensive testing on cross validation data, we found that n-grams actually achieve a lower F1 score than single words (unigrams). We hypothesize that this is because the training set is relatively small, and as a result many n-grams, which convey uncertainty in the test set, will be unseen in the training set.