Natural Language Processing: Assignment 4

Johannes Gontrum

1 Lexical Semantics

1.1 Error Analysis

1.1.1 Word: *hard*

To determine which sense of the word hard is most difficult one to detect, one has to convert the absolute numbers in the confusion matrix into relative values. I did this by dividing all counts by the sum of their row. This way, I can now read the percentage of how often a reference word was classified as a certain sense.

This preprocessing step is necessary because each reading has a different overall frequency. HARD_1 might be misclassified as HARD_2 39 times, but since HARD_1 occurs 702 times in the test set, it a less severe error as the misclassification of HARD_3 as HARD_2 $(12/72)$, which happens 15.6% of the time. Both matrices can be found in Table 1. Using context features, the classifier reached an accuracy of 0.8950.

The sense HARD_3 was classified correctly in only 77.9% of the cases, making it the most difficult sense to detect. Generally, all three senses only differ in the type of nouns they refer to: HARD_1 describes e.g. a difficult task, HARD_2 a dispassionate person and HARD3 an object out of a steady material. During training with a context window of length 3, the classifier can only learn to distinguish the sense by learning the exact nouns that cooccur with a sense but is not able to form any abstraction. I also noticed that the accuracy of each sense is getting worse the less frequent it is, as it seems to mistake a rare sense with a more common one. I could imagine that we would find a different distribution, if we would use the same number of training examples for each sense. Also, annotating the cooccurring nouns with semantic properties like "animate/inanimate" could improve the precision of the model.

1.1.2 Word: *interest*

For the word interest, using the word features turned out more promising with an accuracy of 0.6308 whereas the context features only resulted in a score of 0.4283. Table 2 and Table 3 shows the confusion matrix with relative and absolute scores.

Here, the sense INTEREST_2 performed the worst: From all three testing instances, none was classified correctly, while two of them were confused with INTEREST_6. Both meanings are different: While INTEREST_2 describes something that is causing attention as in "they said nothing of great interest", INTEREST_6 is used in financial terms like "how much interest do you pay on your mortgage?". It is also worth noticing that the INTEREST_6 is with 255 occurrences much more frequent.

The used word features work by taking the 300 most frequent words (without tags) in the document into account, after removing stopwords. As INTEREST_6 is used in the financial context, is not surprising why the classifier mistakes two of the INTEREST_2 examples with them, as they contain more formal and business words like "supreme court ruling", "law" or "finance". It is interesting, however, that when using the context window, the majority of INTEREST_6 words are confused with INTEREST_2, whereas the latter is classified with an accuracy of 1.0.

	HARD_1 (absolute)	HARD_2 (absolute)	HARD_3 (absolute)	HARD_1 (relative)	HARD_2 (relative)	HARD_3 (relative)
HARD_1	643	**39**	20	0.91595	0.05556	0.02849
HARD_2	6	73	9	0.06818	0.82955	0.10227
HARD_3	5	12	60	0.06494	**0.15584**	0.77922

Table 1: Confusion matrix for the three senses of the word HARD with absolute and relative numbers. Rows represent the correct class, columns the predicted class.

	INTEREST$_1$ (absolute)	INTEREST$_2$ (absolute)	INTEREST$_3$ (absolute)	INTEREST$_4$ (absolute)	INTEREST$_5$ (absolute)	INTEREST$_6$ (absolute)
INTEREST$_1$	13	12	0	2	5	**46**
INTEREST$_2$	0	0	1	0	0	2
INTEREST$_3$	0	0	5	6	1	3
INTEREST$_4$	1	9	2	9	7	6
INTEREST$_5$	0	9	3	1	58	18
INTEREST$_6$	0	26	1	1	13	214

Table 2: Confusion matrix for the three senses of the word INTEREST with absolute counts. Rows represent the correct class, columns the predicted class.

	INTEREST$_1$ (relative)	INTEREST$_2$ (relative)	INTEREST$_3$ (relative)	INTEREST$_4$ (relative)	INTEREST$_5$ (relative)	INTEREST$_6$ (relative)
INTEREST$_1$	0.17	0.15	0.00	0.03	0.06	0.59
INTEREST$_2$	0.00	0.00	0.33	0.00	0.00	**0.67**
INTEREST$_3$	0.00	0.00	0.33	0.40	0.07	0.20
INTEREST$_4$	0.03	0.26	0.06	0.26	0.21	0.18
INTEREST$_5$	0.00	0.10	0.03	0.01	0.65	0.20
INTEREST$_6$	0.00	0.10	0.00	0.00	0.05	0.84

Table 3: Confusion matrix for the three senses of the word INTEREST with relative frequency. Rows represent the correct class, columns the predicted class.

1.1.3 Word: *serve*

Using the context window features again, the classifier for the four senses of ”serve” archives an accuracy of 0.8345. Here, SERVE$_2$, meaning to do duty or hold offices, was the most difficult sense to classify with an accuracy of only 0.7929. It was most often mistaken by SERVE$_{12}$, describing someone being adequate, either in quality or quantity as in ”nothing else will serve”. The confusion matrices are shown in Table 4.

I believe the reason for this misclassification lies again in the similar context of both words. The context window features do not filter out stop words, and both senses often occur around the words ”to”, ”as” or ”a”:

”useful allies , SERVIG$_{12}$ both as scouts”

vs.

”and able to SERVE$_2$ as a net importer”

Also, to distinguish between both senses, I think it is important to know the object to which ”serve” refers, but given a window size of 3, it is seldom included. To prevent this, I recommend increasing the window size or including words annotated with their dependency label.

	SERVE$_{10}$ (absolute)	SERVE$_{12}$ (absolute)	SERVE$_2$ (absolute)	SERVE$_6$ (absolute)	SERVE$_{10}$ (relative)	SERVE$_{12}$ (relative)	SERVE$_2$ (relative)	SERVE$_6$ (relative)
SERVE$_{10}$	311	19	09	**32**	0.84	0.05	0.02	0.09
SERVE$_{12}$	11	213	19	4	0.04	0.86	0.08	0.02
SERVE$_2$	2	28	134	5	0.01	**0.17**	0.79	0.03
SERVE$_6$	4	4	8	73	0.04	0.04	0.09	0.82

Table 4: Confusion matrix for the three senses of the word SERVE with absolute and relative numbers. Rows represent the correct class, columns the predicted class.

	Precision	Recall	F_1
HARD_1	0.92	0.98	0.95
HARD_2	0.83	0.59	0.69
HARD_3	0.78	0.67	0.72
Overall	**0.84**	**0.75**	**0.79**

Table 5: Precision, recall and F_1 for the three senses of the word *hard*.

1.2 Precision, Recall & F1-Score

The measurements precision and recall are used to show two different perspectives on the quality performance of a system.

Assuming we have only two classes (match and no match) accuracy alone would give us a high score, even if the system would predict every single word to be a match. Recall, on the other hand, counterbalances this by punishing false negatives. Recall in itself is not enough to evaluate the performance of a system, the F-measure is then used to combine them. F_1 is a special case, where both precision and recall are weighted equally.

Precision is formally calculated as shown in Equation 1, as defined by Manning et al. (1999). Informally it is the number of true positives divided by the sum of the row. Recall (see Equation 2, Manning et al. (1999)) in contrast divides by the sum of the column. F_1 is a balanced way of combining both scores into one. It is defined in Equation 3 (see also Jurafsky and Martin (2009)). Precision and recall are defined to be used with binary classification, so we have to compute the scores for each sense individually (see Table 5).

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}} \tag{1}$$

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \tag{2}$$

$$F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3}$$

Averaging the scores, we end up with a precision of 0.8416, a recall of 0.7487 and an F_1 measure of 0.7866.

2 Semantic role labeling

2.1 Annotation

We discussed our annotations of the given ten sentences in a group of four, while two of us did only annotate senses, but not spans or roles. During our discussion, we discovered that the sense annotation was by far the easiest part. We annotated most verbs exactly the same and had only a few discussions which could easily be resolved with the sense definitions.

Annotating the roles of the arguments turned out to be a bit more difficult. In several senses, the verb 'count' was used in passive, which means that we would not annotate any object as ARG0. Not everyone in the group agreed with this interpretation, but we decided to change our combined annotation accordingly.

The biggest confusion was about the correct annotation of the spans of the arguments. In my annotation, I tried to keep them as short as possible to not include any determiners or other words that might belong to the phrase. The majority of the group, though, disagreed with this and we preferred longer spans in the end. All in all, our approach was to keep consistent with our way of annotating, even though we were tempted to change our minds later on.

2.2 Inter-annotator agreement metrics

There exist several metrics to measure how good multiple coders agree on their annotations. The easiest one is to simply calculate the percentage of items that the annotators have agreed on. However, this naive metric has several flaws: Artstein and Poesio (2008) argue for example that percentage agreement has higher scores, the fewer the number of categories that are used. Also, it does not take into account that annotators might only agree by chance.

	Judge 2 (Group)				
Judge 1 (me)	**count.01**	**count.02**	**count.03**	**count.04**	**Total**
count.01	4	0	0	0	4
count.02	0	3	0	0	3
count.03	0	0	2	0	2
count.04	0	0	0	1	1
Total	4	3	2	1	10
Agreement	4	3	2	1	10
By chance	1.6	0.9	0.4	0.1	3
Kappa (κ)	1				

Table 6: Calculation of the κ metric for the inter-annotator agreement between my annotation and the group annotation.

Cohen's κ, on the other hand, belongs to the category of chance-corrected metrics that integrate the probability that two annotators coincidentally agree. In Table 6, I calculated the agreement between my annotation of the four different senses of the count and the annotation we agreed on after the mentioned group discussion. Even though we agree in 100% of the cases, resulting in a κ score of 1, the row 'By chance' demonstrates how the metric works. For each sense, the total number of each annotator is multiplied and the result divided by the overall number of items (here: 10 sentences). To calculate the κ score, one has to subtract the sum of the 'By chance' row from the number of all agreed senses and then divide by the total number of items minus the summed up chance scores.

In recent years, researchers have questioned whether κ is a good way to measure inter-annotator agreement because it underestimates rare categories. Others believe that the calculation of the probability that annotators agree by chance is too naive and are concerned because κ does not take a shared annotates bias into account. As a consequence of this discussion, Krippendorff's α, that addresses some of these concerns, is more often used.

References

Ron Artstein and Massimo Poesio. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596, 2008.

Daniel Jurafsky and James Martin. *Speech and language processing: An introduction to natural language processing*. Prentice Hall, 2009.

Christopher D Manning, Hinrich Schütze, et al. *Foundations of statistical natural language processing*, volume 999. MIT Press, 1999.