Text Mining - Assignment 2 - Sequence Labelling

Andreas Savva
s3316491

Ivan Horokhovskyi
s3069176

1 Introduction

In this assignment, we performed sequence labelling by applying conditional random fields. We have created a baseline (inspiring from CRF scikit-learn tutorial[1]), introduced new features, tuned models via hyperparameter optimisation (HPO) and compared the results along with introducing the most impactful feature combinations.

2 Conditional Random Fields

The predictor we are using is **C**onditional **R**andom **F**ield which is a supervised conditional machine learning model with Limited BFGS as optimizer. CRF is a discriminative sequence model which uses feature vectors and the previous context. Rather than trying to compute the probability for a tag, it computes the global probability of the sentence.

$$P(\hat{y}|\hat{x}; w) = \frac{exp(\sum_i \sum_j w_j f_j(y_{i-1}, y_i, \hat{x}, i)}{\sum_{y' \in Y} exp(\sum_i \sum_j w_j f_j(y'_{i-1}, y'_i, \hat{x}, i))} \tag{1}$$

In equation [1], $\hat{y}$ is the sequence of tags we predict, $\hat{x}$ is the sequence of words, w are the weights. So the numerator uses the feature function f_j which uses the y$_{i-1}$, the previous context, y_i current context and the sequence, $\hat{x}$. The feature function is multiplied by the weight of the feature w_j and all features functions are summed with $\sum_j$ which in turn is summed for all of the sequence lengths $\sum_i$. In the denominator, we use the same as the exponent but sum over all possible label sequences, $\sum_{y' \in Y}$.

3 Dataset

The dataset is taken from the Workshop on Noisy User-generated Text (W-NUT) [2].

There are 6 entity labels in the dataset namely: Person, Location, Corporation, Proudct, Creative work, Group. Since these entities can be constructed as 2 or more words, the dataset provides 2 labels from each of the aforementioned labels, 'B' for the first token of the entity and 'I' for all the following tokens. In Table 1 the distribution of

Label	Train	Dev	Test	Total
B-person	660	470	429	**1559**
B-location	548	74	150	**772**
I-person	335	117	131	**583**
I-creative-work	206	133	218	**557**
B-group	264	39	165	**468**
I-product	203	94	126	**423**
B-creative-work	140	105	142	**387**
B-product	142	94	127	**383**
I-location	245	33	94	**372**
B-corporation	221	33	66	**320**
I-group	150	25	70	**245**
I-corporation	46	11	22	**79**

Table 1: Number of occurrences of each label (distribution) in train, dev, test and the entire dataset

values per each label in train, dev and test sets are present. As can be observed the label distribution is slightly different in train, dev and test which can lead to worse performance. In Figure 1 the distribution for sentence lengths is provided, it can be seen that the test set has a long distribution tail.

The dataset has 5689 records in total which were split into the train, validation (dev) and test in 60%, 23% and 17% respectively. The dataset itself is kind of small, which is why it is important to enrich it with additional features, but simultaneously try not to overfit the model.

4 Experiments

To make the experiments reproducible a random state was fixed. All the experiments were made on python 3.10 (scikit-learn=0.22.1, nltk=3.7, spacy=3.4.3).

The dataset comes tagged with BIO tags. We tag the dataset with POS tags. As base features, we have the lowercase word, character bigrams/trigrams of it, if it is an uppercase/title/number and the pos tag. After inspecting the dataset we have created additional features for the CRF classifier. We have noticed that there are quite some hashtags "#", mentions "@" and punctuation signs (including repeating signs). There are also some URLs (both with

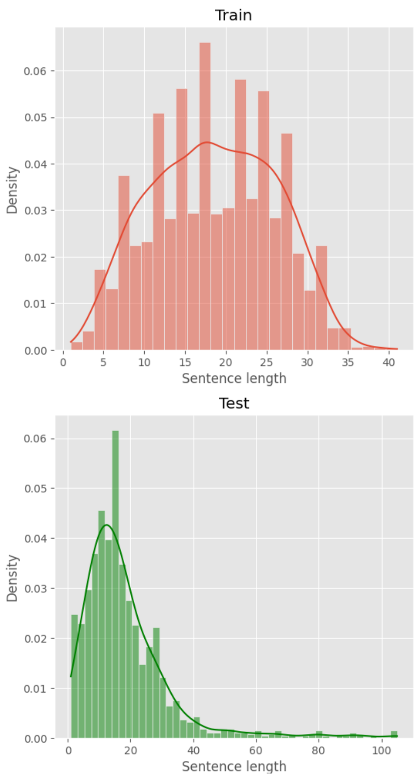

Figure 1: Dataset sentences length distribution for train and test sets

http(s) and www formats), and some tokens representing money. For all aforementioned, we have created regular expressions and made boolean features which check if a word matches the expression. In addition, we have applied the common NLP techniques of stop words check and applied lemmatization. These techniques are used as a feature, so for a word, we have a feature representing its lemma and one showing whether it is a stop word. Last but not least, we tried to use the Word2Vector model[3]. For the Word2Vec models we tried training our own model on the train set sentences using vector size of 5, we used a pre-trained model (glove-twitter-25 [4] [5]) with vector size 25 and also tried a finetuned model combination where we used the pre-trained model and and our own model with vector size 25. Since the CRF library does not support lists as features, the embeddings had to be represented as individual features which is not ideal.

HPO was made with a random search algorithm and cross validation. The validation set in combination with the train set was used to perform the hyperparameter optimization. In HPO, we use cross-validation using both sets but entries from the train set are only available in the training split and in the test, whereas entries of the dev set can be in both. Random search remains simple in implementation and converges a lot faster in comparison to grid search. Cross-validation allows us to receive more statistically significant results for HPO.

As for experiments: we have tried all the features individually to see which features contribute the most. And after that, we started to combine features. The big challenge was not to overfit the model which was easy taking into consideration that the dataset is small, we can potentially add a lot of features and the dataset has previously-unseen entities.

5 Results

As the baseline, we have taken the tutorial process and generated the scores. Before HPO the F1 score is 0.13 and after that, it raises up to 0.19. The baseline scores are a great example of how HPO impacts the metric values. The F1 score for the train set often achieved a value of 0.99 which is a big difference from the test score, which may signal about overfitting. All the following results are given after HPO only.

In Table 2 the results can be observed for individual custom features. In general, results are predictable, except for the *is money* & *has emoji* features, which can be probably explained that they are pretty rare but accurate features so the model does not overfit but benefits from them. As individual features, we would still highlight: *is first capital*, *is hashtag*, *word2vector custom* and *has emoji*.

Feature	**Precision**	**Recall**	**F1**
has-emoji	0.431	0.168	0.230
is-first-cap	0.386	0.165	0.221
is-hashtag	0.412	0.164	0.223
is-mention	0.413	0.156	0.211
is-money	0.428	0.169	0.230
is-punct	0.395	0.155	0.213
is-rep-punct	0.366	0.138	0.187
is-url	0.397	0.154	0.211
is-stopword	0.417	0.155	0.214
lemma	0.373	0.149	0.204
w2v-custom	0.390	0.164	0.220
w2v-pretrained	0.328	0.145	0.195
w2v-finetuned	0.385	0.145	0.199

Table 2: Results for custom features when tested after hyperparameter optimization. Uses the base features with context size = 2. Custom features do not have context.

Feature	Precision	Recall	F1
avg	0.41	0.16	0.22
B-person	0.59	0.30	0.39
I-person	0.51	0.28	0.36
B-location	0.43	0.28	0.34
I-location	0.41	0.16	0.23
I-product	0.30	0.12	0.17
I-corporation	0.50	0.09	0.15
I-creative-work	0.32	0.09	0.14
B-corporation	0.67	0.06	0.11
B-creative-work	0.31	0.06	0.10
B-product	0.30	0.06	0.09
B-group	0.26	0.03	0.05
I-group	0.11	0.01	0.03

Table 3: Results applying finetuned word2vec after hyperparameter optimization. With context size = 1 for word2vec features and context size = 2 for basic features.

Feature	Precision	Recall	F1
avg	0.41	0.17	0.22
B-person	0.59	0.32	0.42
I-person	0.51	0.32	0.39
B-location	0.41	0.29	0.34
I-location	0.31	0.16	0.21
I-creative-work	0.46	0.09	0.15
I-group	0.35	0.09	0.14
B-creative-work	0.40	0.06	0.10
I-product	0.23	0.06	0.09
B-product	0.24	0.04	0.07
B-corporation	0.40	0.03	0.06
B-group	0.17	0.03	0.05
I-corporation	0.00	0.00	0.00

Table 4: Results applying all features (uses fine-tuned pretrained model for embeddings using our custom Word2Vec model) after hyperparameter optimization, without context on custom features.

Due to the page limitation of the assignment, we can't show all the experiments we have done. In all experiments, there is a clear trend that precision values are much higher than recall values. From Tables 3 and 4 it can also be observed that *I* labels tend to have higher recall values which in the end gives them higher F1 values. This fact can be probably explained that entities which consist of 2 and more words are easier to be spotted by the model. Moreover, based on our features it is easier to identify persons and locations since they have the highest metrics values. Another curious thing is that for the model with all features (Table 4) *I-corporation* value is 0.

6 Conclusions

In this assignment, we have experimented with sequence labelling, worked with conditional random fields model, constructed additional features and performed hyperparameter optimization. The big deal breaker for receiving better metrics was the dataset, to be precise the size and train/dev/test split. In the end, the recommended set of features is finetuned word2vector features along side the base features because in the case of a larger dataset it can be assumed that the model will overfit less and get the full benefit of its features. Furthermore, adding some more trivial features like *is mention*, *is hashtag*, *is first capital* is beneficial.

CRF is a lightweight and simple model. However to get better accuracy more advanced models like BiLSTM-CRF or BERT can be used.

References

[1] Scikit-Learn CRFsuite tutorial. `https://sklearn-crfsuite.readthedocs.io/en/latest/tutorial.html`.

[2] Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[3] Efficient Estimation of Word Representations in Vector Space. `https://arxiv.org/pdf/1301.3781.pdf`.

[4] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[5] GloVe: Global Vectors for Word Representation Repository. `https://nlp.stanford.edu/projects/glove/`.