Feature Analysis for Named Entity Recognition

Gaurav Rajesh Parikh

March 6, 2024

For this task, I choose to put all my effort on improving and selecting good features rather than constructing Neural Network Architectures. This decision was made since the dataset is small which means it might not be able to train a neural network very effectively. I also placed emphasis on interpretable Machine learning features (which is my strength) and picked out selected textual features that are effective predictors for entities. I also note that some features I select might even be noise but they did help in preventing overfitting.

- **Current Word**: I selected this feature to represent the current word in the text sequence. It provides direct information about the word being analyzed.
- **Previous Word and Next Word**: These features capture the context of the current word by considering the preceding and succeeding words. Contextual information helps in disambiguating the meaning of the current word.
- **Start Characters**: The features *isupper* and *islower* indicate whether the current word starts with an uppercase or lowercase letter, respectively. I used this information as I felt it was useful for identifying proper nouns.
- **Part-of-Speech (POS) Tag**: The POS tag of the current word provides syntactic information about its grammatical category (e.g., noun, verb, adjective). I picked this feature as it helps in capturing the word's linguistic properties. Nouns are more likely to be "proper nouns" and thus places or persons.
- **Common Noun Indicator**: This binary feature (*is_common_noun*) indicates whether the current word is a common noun. It is based on WordNet synsets and can assist in distinguishing named entities from common nouns. I used this external corpus to enrich model knowledge. We were told that the train and test set have domain differences, so I used a general corpus, otherwise I would have simply taken the top K words with highest frequency, manually labelled them as being common nouns and then added them to context.
- **Digit and Punctuation Presence**: These features (*has_digits* and *has_punctuation*) indicate whether the current word contains digits or punctuation marks, respectively. They can aid in identifying numeric entities and punctuation's and can be useful in identifying place names such as "NC, USA".
- **Prefix and Suffix**: These features capture the first three characters (prefix) and last three characters (suffix) of the current word. Prefix and suffix information can help in identifying word morphology and patterns. The idea was to capture more lexical information.
- **Relative Position in Sentence**: This feature (*rel_position*) represents the relative position of the current word in the sentence. It normalizes the position based on the sentence length and can assist in understanding word importance.
- **Distance from Last Start Word**: This feature (*distance_from_last_s*) measures the distance of the current word from the last observed start word (<s>). It provides sequential information and can help in detecting sentence boundaries under the assumption that class distribution is not uniform across sentence position.
- **Stem Token**: This feature (*stem_token*) represents the stemmed form of the current word. I used this as stemming reduces words to their base or root form, which can improve generalization by treating similar words as the same entity.
- **Word Length**: This feature (*word_length*) indicates the length of the current word. I used this as word length can be indicative of entity types, such as short words being more likely to represent named entitie such as organizations :"UN, WTO, EU"

1 Conclusion

By effectively selecting features, and running a variety of experiments I was able to obtain **validation F1 scores of 0.87 which gave me a test set F1 of 0.746 as opposed to the 0.702 I obtained from the original CRF.**

Generally, I had higher precision than recall suggesting that entities might not be as well detected but when detected, the model is reliable. After achieving a good performing model based on feature engineering, I lowered the learning rate to allow for slower convergence that might be able to find a better minima and added early stopping with patience (to account for local non convexity in the loss landscape) and increased the number of epochs to train my model better. I also included gradient max norm clipping to prevent gradient explosions under this paradigm to finally achieve my best results.

Table 1 also shows how much adding some selected features improved the F1 Score. Plot 1 shows best model epochs and Precision, Recall and F1 score.

Table 1: Model Performances

Feature	**Best Validation F1 Score**	**Best Test F1 Score**
Base Model	0.834	0.706
Base Model + Prefix/ Suffix	0.849	0.734
Base Model+ Prefix/Suffix+ Stemming	0.855	0.736
Base Model+ Prefix/Suffix+ Stemming+ POS tagging	0.856	0.743
Final Model (aforementioned + some additional features)	0.87	0.746

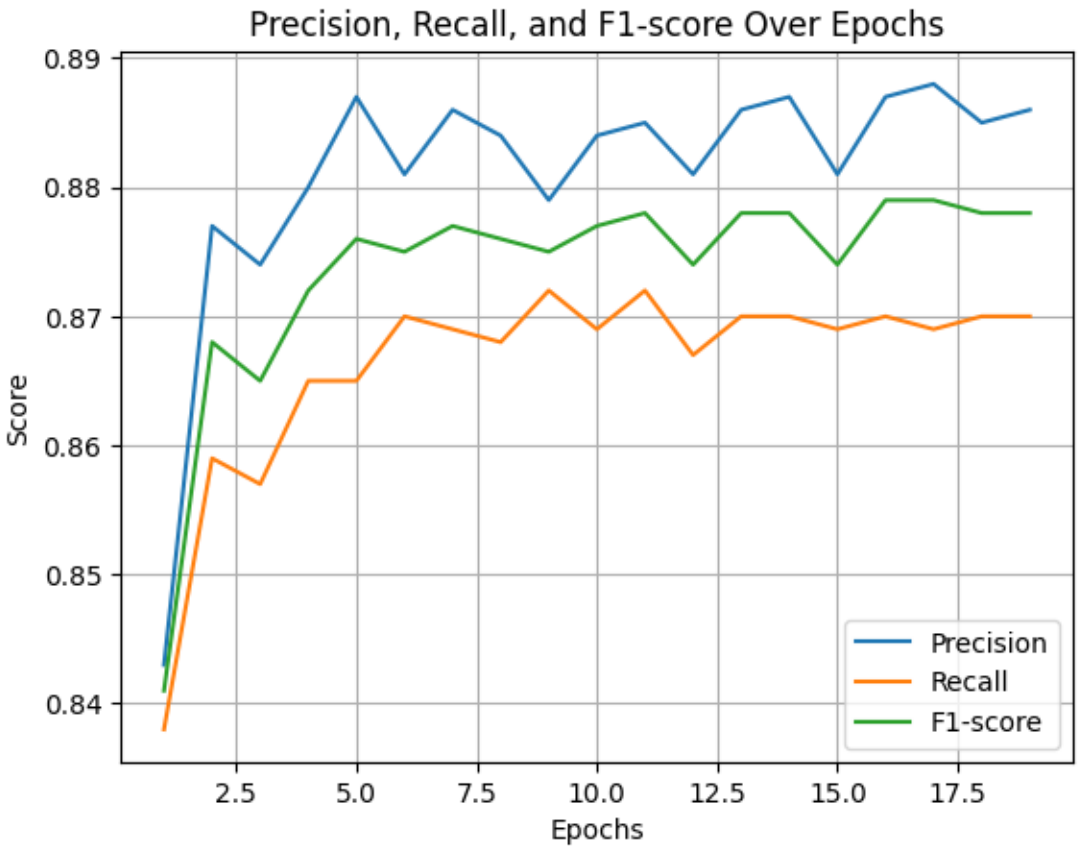

Figure 1: Best Model Validation Performance over 20 epochs