Report

Named Entity Recognition

Team Name: Cheese Maggi

Aryan Jain(2019101056) **Palash Sharma(2019101082)** **Arth Raj(2019101094)**

Problem Description: NER is an information extraction technique to identify and classify named entities in text. These entities can be pre-defined and generic like location names, organizations, time etc, or they can be very specific like the example with the resume.
Snorkel is a system that facilitates the process of building and managing training datasets without manual labeling. The first component of a Snorkel pipeline includes labeling functions, which are designed to be weak heuristic functions that predict a label given unlabelled data.

Motivation: NER has a wide variety of use cases in the business. For instance Gmail is applying NER when you are writing an email and you mention a time in your email or attaching a file, gmail offers to set a calendar notification or remind you to attach the file in case you are sending the email without an attachment. Other applications of NER include: extracting important named entities from legal, financial, and medical documents, classifying content for news providers, improving the search algorithms, and etc.

Literature Review: We went through several research papers and articles to understand properly how the NLP world has progressed on the problems of annotation. Some of the papers/links we referred are:

1. **Named Entity Recognition without Labeled Data: A Weak Supervision Approach:** This presents a simple but powerful approach to learn NER models in the absence of labeled data through weak supervision. The approach relies on a broad spectrum of labeling functions to automatically annotate texts from the target domain.

2. **Snorkel: rapid training data creation with weak supervision:** They present a flexible interface layer for writing labeling functions based on their experience over the past years collaborating with companies, agencies, and research labs.

3. **Labeling Data Using Snorkel:** We referred to an online link from KDnuggets, where they have walked through the process of using Snorkel to generate labels for an unlabelled dataset. In this article they have demonstrated label generation for MS severity scores.

4. **Snorkel Intro Tutorial: Data Labeling:** This article is from snorkel's official website which describes the process of using Snorkel to build a training set for classifying YouTube comments as spam or not spam. They illustrated the basic components and concepts of Snorkel in a simple way, and also showed the actual process of iteratively developing real applications in Snorkel.

5. For further deep understanding of the NER we referred to a paper on the same by Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li, which is **A Survey on Deep Learning for Named Entity Recognition.**

Dataset Used:

For Snorkel labelling: We have used re3d defense dataset for our NER extraction. The data is simply an English language text. Words of each sentence are broken down and written line by line in the file along with its corresponding NER tag. The end of the sentence is marked by an empty line. We studied data and found that IOB [Intermediate-Out of tag-Beginning] tagging is done on the dataset with the entities being: organization, quantity, weapon, documentreference, person, nationality, money, temporal, location, military platform. The dataset is divided into train-validation-test files.

For training Neural Network: We have used the Conll2003 News dataset. The format of this dataset is exactly the same as the one used during Snorkel labeling. Just that here we have entities: organization, location, miscellaneous, person.

Architecture: The architecture is designed in three steps. Firstly we build labelled data using weak supervision with the help of snorkel and then in next step we pass this data to a pretrained model(which is trained in some other domain). After this we finetune(transfer learning via bootstrapping) it which results in significant improvements in accuracy and F1 score.

Weak Supervision using Snorkel:

To label unlabelled data, we can turn to a *weak supervision* approach, using ***labeling functions (LFs)*** in Snorkel: noisy, programmatic rules and heuristics that assign labels to unlabelled training data. We have used 8-10 labeling functions for the re3d dataset to classify them into classes: Money, Quantity, Document Reference, Location, Weapon, Nationality, Organization, Temporal. The tokens which do not fall in any of these classes are labeled as ‘O’, which means Out of Tag. Including the above classes, the data is also tagged as I/B i.e. we have tagged a token as b-money if it marks the starting of a money token and i-money if it is some intermediate money token.

Labeling functions: The labeling function we have used to label the data are:

- **Money labeling Function:** This labeling function marks the keywords as money which are related to money. We looked for the keywords like '$', '£','wage', 'wages','salaries' etc. to mark them as b-money and for ‘'dinars', 'dollars', 'millions', 'donor', 'jordanian', 'billions’, we marked them as i-money.

- **Quantity labeling Function:** This labeling function marks the keywords as quantity which are related to quantity of something . We looked for the keywords like 'percent', 'miles', 'domains', ,'millions', 'thousands' ,'kms', 'squares', 'acres’ etc. and marked them as quantity.

- **Document Reference labeling Function:** This labeling function finds the keywords which have reference to some document. Some of the keywords for which have looked are: ‘books’, ‘chapter’, ‘resolution’, ‘texts’, ‘reports’, ‘constitution’, ‘united nations’, ‘executives’ etc.

- **Location labeling Function:** This labeling function is used to mark location names mentioned in the data. For this we looked for the keywords ‘in’, ‘near’, ‘above’, ‘over’, ‘by’, ‘along’, ‘around’ and marked them as b-location. The keywords following these would be actually some location, which we marked as i-location.

- **Weapon labeling Function:** This labeling function marks all the keywords related to weapons. Example of such keywords are: ‘weapons’, ‘strikes’, ‘rocket’, ‘chemical’, ‘missile’, ‘bomb’ etc.

- **Nationality labeling Function:** This labeling function tags all the tokens which are related to nationality, nations, or keyword referring to a group. Examples of such keywords are ‘Arabic’, ‘Iraqi’, ’Australians’, ‘Muslims’, ‘Northern’ etc.

- **Organization labeling Function:** This labeling function marks the keywords which are related to organization or some group. The keywords used to identify such groups are: ‘forces’, ‘Coalition’, ‘United’, ‘People’, ‘regime’ etc.

- **Temporal labeling Function:** This labeling function is used to mark the keywords related to time. Examples of such keywords are: ‘year’, ‘hours’, ‘minute’, ‘days’, ‘week’, ‘december’, ‘day’, ‘last’, ‘past’, ‘months’ etc.

After labeling the data using the above mentioned labeling functions, our goal is now to convert these labels from our LFs into a single *noise-aware* probabilistic (or confidence-weighted) label per data point. For this we have used sophisticated Snorkel **LabelModel** to combine the outputs of the LFs. The label model function requires a label matrix which is simply a NumPy array L with one column for each LF and one row for each data point, where L[i, j] is the label that the jth labeling function outputs for the ith data point. This label matrix is passed to the Label Model function, along with the number of epochs(1000 in our case) to fit to the given data and produce single noise aware labels.

Neural Model:

The neural model is the second component of the project architecture. The first component takes human intervention to know the hard rules to label a dataset. This part of the architecture tries to learn the intuition from this weak labeling to label the same dataset with a much better accuracy.

The main architecture of the neural model consists of:

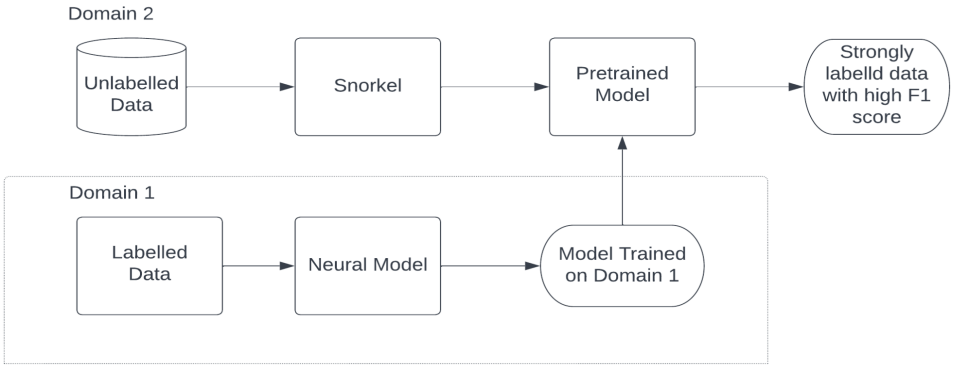
Layer 1: *Elmo Pretrained embedding model*
Layer 2: *Bi-LSTM with dropout*
Layer 3: *Bi-LSTM with dropout*
Layer 4: *Fully connected layer with softmax activation*

This model is trained on a sufficiently large dataset first so that the inherent nature of the problem and understanding required is gained by the LSTM layers.

Transfer learning via Bootstrapping (Fine-tuning):

This is the last part of the architecture where we need the pre-trained neural model to learn new domain specific information. Hence, we remove the last layer and replace it with 2 new FC layers, freeze the old layers and train the model on new, weakly labeled (by snorkel) dataset so that the new layers learn how to classify the new tags needed. Finally, all the layers are unfrozen and fine-tuned on 100-200 epochs to get a supreme F1-score.

Pipeline:



Hyperparameters

Model
Optimizer :- Adam optimizer involves a combination of two gradient descent methodologies:
Momentum: This algorithm is used to accelerate the gradient descent algorithm by taking into consideration the 'exponentially weighted average' of the gradients.

Sparse_categorical_crossentropy :- Used as a loss function for multi-class classification models where the output label is assigned integer value (0, 1, 2, 3…). This loss function is mathematically the same as the categorical_crossentropy. It just has a different interface.

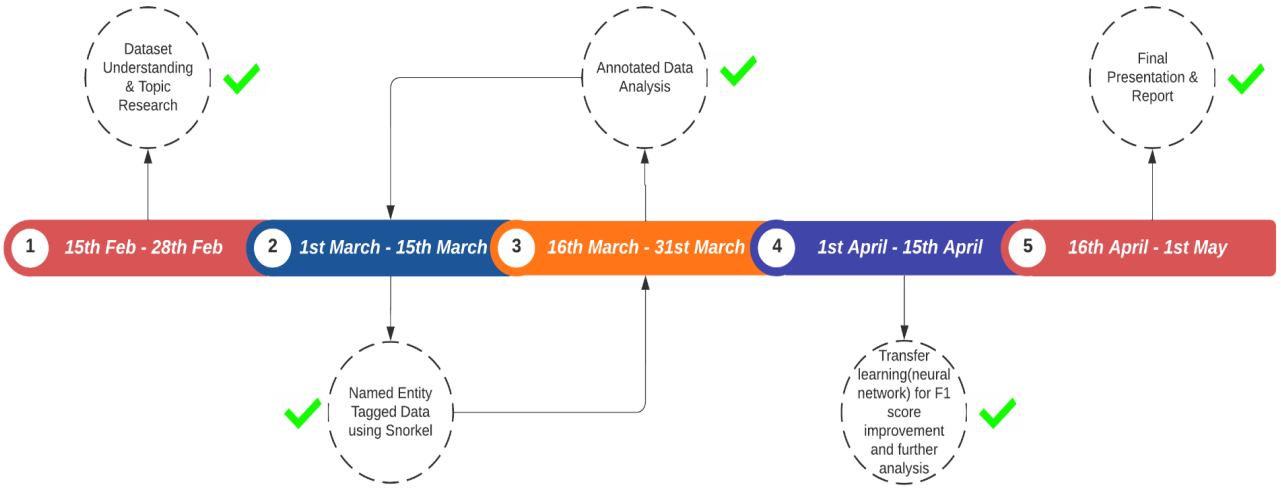
Accuracy metric :- Various accuracy metrics like accuracy, F1 score and loss were calculated for better track of learning

Epochs :- 3 were used for initial neural model training, while 3 and 100 were used for finetuning on the snorkel labeled dataset

LSTM
Units :- 512
The output shape of the final dense layer will be affected by the number of neuron / units specified.

Dropout :- 0.2
Every LSTM layer should be accompanied by a dropout layer. Such a layer helps avoid overfitting in training by bypassing randomly selected neurons, thereby reducing the sensitivity to specific weights of the individual neurons.

Timeline:



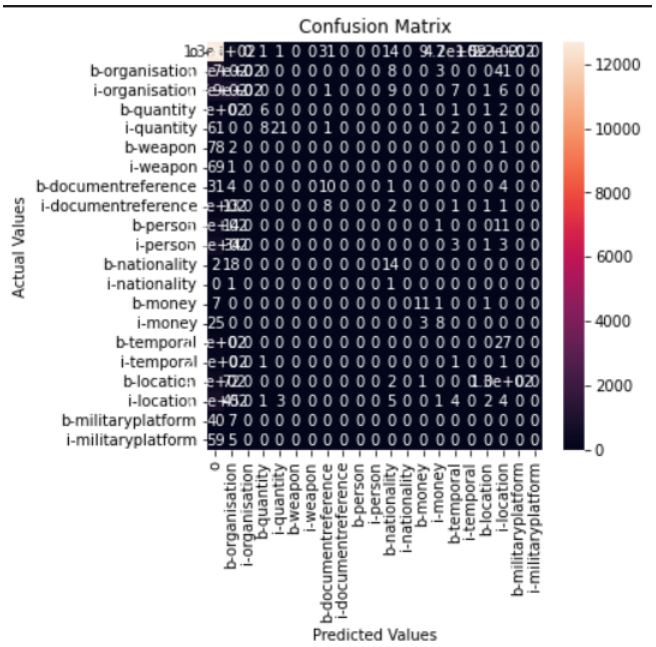
Results & Analysis:

Snorkel Results:

	precision	recall	f1-score	support
o	0.73	0.92	0.82	13747
b-organisation	0.18	0.24	0.21	1104
i-organisation	1.00	0.00	0.00	1539
b-quantity	0.35	0.04	0.07	150
i-quantity	0.84	0.22	0.35	94
b-weapon	1.00	0.00	0.00	81
i-weapon	1.00	0.00	0.00	70
b-documentreference	0.20	0.20	0.20	50
i-documentreference	1.00	0.00	0.00	146
b-person	1.00	0.00	0.00	411
i-person	1.00	0.00	0.00	605
b-nationality	0.25	0.41	0.31	34
i-nationality	1.00	0.00	0.00	2
b-money	0.44	0.55	0.49	20
i-money	0.38	0.22	0.28	36
b-temporal	0.00	0.00	0.00	177
i-temporal	1.00	0.00	0.00	225
b-location	0.00	0.00	0.00	644
i-location	0.01	0.01	0.01	784
b-militaryplatform	1.00	0.00	0.00	47
i-militaryplatform	1.00	0.00	0.00	64
accuracy			0.65	20030
macro avg	0.64	0.13	0.13	20030
weighted avg	0.68	0.65	0.58	20030

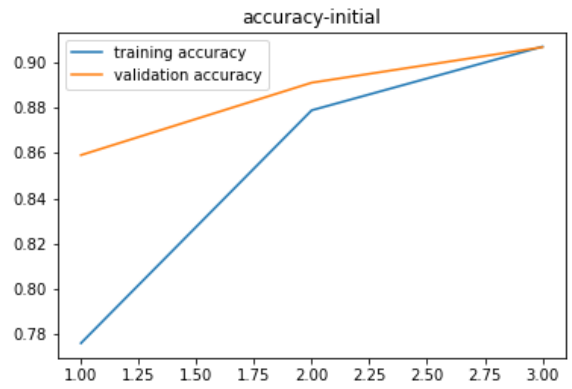
We ran our snorkel based weak supervision approach on re3d data and obtained an accuracy of ~65%. Cumulative F1 score came out to be around 0.58.

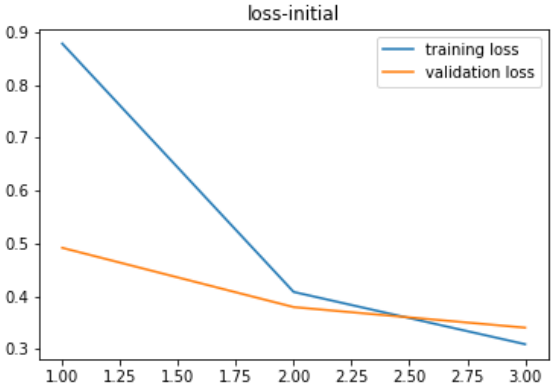
The image on the left is confusion matrix plotted for all the labels for the ner labeling task for the given defense data

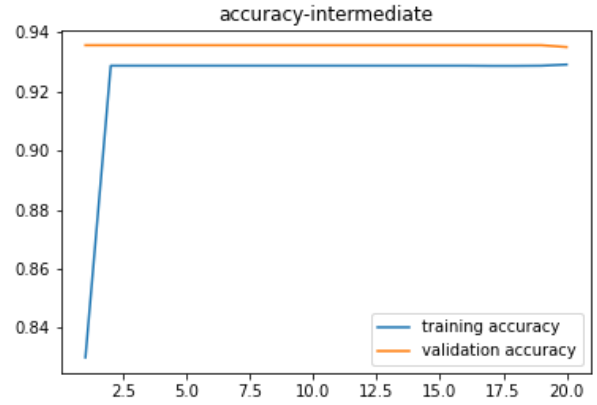


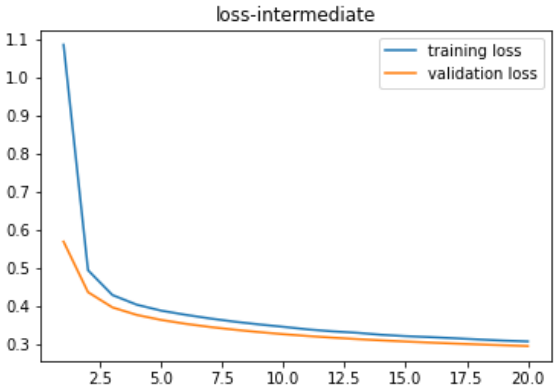
Confusion Matrix obtained from the snorkel labelled data is:

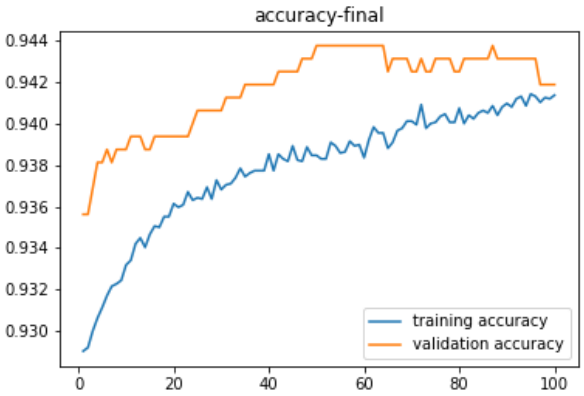
Neural Model training and transfer learning results

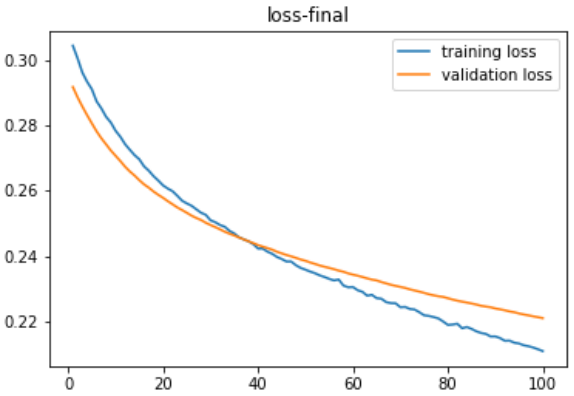












It is visible above that the accuracies have reached a good value at the end of each training for all training steps involved in the architecture. Accuracies upto 96% were reached consistently.

Conclusion

Manual labeling has been a big problem in the field of Machine Learning. To solve this problem, one can label any unlabelled dataset, firstly using weak supervision with the help of snorkel and then, the tagged data can be passed to a pre-trained neural model with some fine tuning, results in significant accuracy improvement and high F1-score. This is achieved due to the transfer learning approach used. Hence, unlabelled datasets can be labeled with sufficiently good F1 scores for use in various ML tasks.

Full Implementation is available at: https://github.com/Rockarya/NER