Name Entity Recognition (NER) in Emails

Chenze Li, Tianyi Feng
Department of Statistics, University of California, Davis, CA 95616
czeli@ucdavis.edu,tfeng@ucdavis.edu

Abstract

Name Entity Recognition is a well studied topic that aims at identifying name tokens in a given text. NER in email is especially difficult due to certain characteristics of email corpus. This report focus on meeting email data obtained from Enron Meeting Data set. Three models (Latent Dirichlet Allocation, Random Forest and Conditional Random Fields) are applied to the data set to complete the NER task. The Random Forest model has high precision and low recall. The CRF model has moderate precision and recall. The LDA model has worst performance. CRF models shows some interesting patterns in both name and normal words.

1 Introduction

Name Entity Recognition(NER) is a NLP task that identifies name entities in the text. The names being identified only include human names like 'Mathew', 'Bill' etc. In previous studies, NER has been extended to news articles, scientific articles, web pages and other fields. But in most of the fields, only formal text are included. Email text is a special type of text in that most content in emails are informal. This include incomplete grammar structures in the header field, short in length causing lack of context, isolated name tokens in the 'sign off' area and etc. These features of email text poses nature obstacles for NER.
Another notable aspect of the corpus we use is that they are mostly meeting-related emails. The main topics of this kind of emails are reservations and cancellations of meetings, member lists of meetings. They are typically short since such information do not require a lot of words to illustrate.
In this report, We will focus on this specific type of meeting email text. We will try to implement popular methods in NER like CRF models as well as other machine learning models like Latent Dirichlet Allocation(LDA) and Random Forest. We need to tailor feature set to make it better fit the email text. After models are trained and predictions are made, we need to properly evaluate the results and conclude with the best models.

2 Problems and Data Description

In this report, we will build models to identify name in emails. To achieve this, we acquire our corpora from http://www.cs.cmu.edu/ einat/datasets.html. The data set "Enron-Meetings" we use is extracted from the Enron corpus. This corpora are mostly related to meetings. This data set is split into two parts—train data set and test data set. In each email within this data set, all names are labelled with XML tags.

After completing data pre-process which will be introduced later,for random forest model, the train data contains 610 unique names and 2668 words except name;the test data contains 220 names and 1004 words except name. For Latent Dirichlet allocation, the train data contains 688 unique names and 2111 "Normal" words; test data contains 220 names and 848 "Normal" words.

This report is the final project for STA221 instructed by Xiucai Ding

3 Related Work

NER has been applied to a wide range text scenarios, such as news articles (McCallum and Li, 2003), scientific articles (Bunescu and Mooney, 2004), web pages (Freitag, 1998), and emails (Einat Minkov et al., 2005). A popular method is CRF(Lafferty et al.,2001) especially linear chain CRFs. CRF takes advantage of sequential structures in sentences and they have shown good performance in POS tagging (Lafferty et al., 2001), noun phrase segmentation (Sha and Pereira, 2003) and Chinese word segmentation (McCallum and Feng, 2003). Latent Dirichlet Allocation (David M. Blei, Andrew Y. Ng, Michael I. Jordan, 2003) is a generative probabilistic model for collections of discrete data such as text corpora. It also has wide applications in tag recommendation (Ralf Krestel et al., 2009), identifying word features (J Petterson et al.,2010) and many other fields. Since identifying name tokens can be regarded as tagging the name, LDA is suitable for the NER task. Random Forest Classifier (Leo Breiman et al., 1995) is a classic and efficient machine learning model. RF has been implemented to NER tasks in health records (A Magga et al., 2018). It has also been applied in other NLP tasks such as sentimental analysis (M Kanakaraj et al., 2015), identifying crisis-related messages (M Imran et al., 2016) and others.

4 Proposed Methods

4.1 Intuition

Since there are many mature models for Named Entity Recognition(NER), this report only applies some these models to the email corpora and make some adaptations accordingly. And since many models concentrate on clustering to achieve NER in unsupervised learning, we would like to use Latent Dirichlet Allocation to conduct a rough model to make classifications between name and other texts. Also, we conduct Random Forests which is a supervised learning method to make classifications. Besides, conditional Random Fields is a classical model for NER tasks. Although the typical sentence length of email text may harm the performance of CRF model, we still decide to apply it to the data set.

4.2 Descriptions of Models

4.2.1 Latent Dirichlet Allocation

Latent Dirichlet allocation(LDA) is an unsupervised learning technique for topic modeling and in this report we will use this method to identify names in a document.

LDA assumes that all documents consist of a few of topics and each topic can be described by combinations of words. A corpus is represented as $D = (w_1, w_2, \cdots, w_N)$ with w_i as a document vector. In a document w_i, $w_i = (\beta_{i1}, \beta_{i2}, \cdots, \beta_{iM})$ with β_{ik} as the kth word in document i. Then LDA assumes the following process for the generation of a corpus:

- For a document, choose a variable $\theta \sim Dirichlet(\alpha)$
- For each word β_k in the document:
  - Choose a topic $z_k \sim Multinomial(\theta)$ for this word.
  - Generate a word β_k which is the kth word in the document from $p(\beta_k|z_k, \eta)$, a probability conditioned on the z_k and η which is a estimated parameter.

Figure 1 shows the illustration of this generative process. Based on the above assumptions, the joint distribution of topic parameters θ, z and a document w_i could be shown as:

$$p(\theta, z, w_i|\alpha, \eta) = p(\theta|\alpha) \prod_{k=1}^{M} p(z_k|\theta) p(\beta_k|z_k, \eta)$$

Then for the probability of the corpus D, it could be represented as:

$$p(D|\alpha, \eta) = \prod_{d=1}^{N} \int p(\theta_d|\alpha) \left(\prod_{k=1}^{M_d} \sum_{z_{dk}} p(z_{dk}|\theta_d) p(\beta_{dk}|z_{dk}, \eta) \right) d\theta_d$$

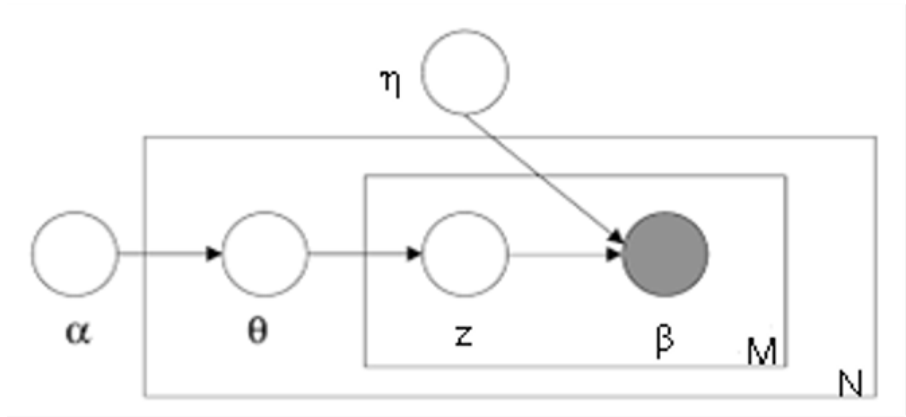

Figure 1: Graphical illustration of LDA

In practice, we only observe the documents and words β_{dk} and we use Bayesian model to compute the posterior distribution of the parameters.

In this report, we first generate a name reference from the training data set and use LDA model to generate K topics based on the training data. Then if word β_{dk} is most relevant with topic k based on the probability, then we say word β_{dk} belongs to topic k. If the majority of names belongs to some topics, then we denote these topics as name topics. For a new corpus, we use the trained model to predict the topics of words in the new corpus and identify names based on the name topics.

4.2.2 Random Forests

Random Forests Algorithm is a supervised learning technique and also a popular and wildly used example of ensemble learning methods. It mainly consists of two parts:bagging or also names as bootstrap aggregating and decision trees. As for bootstrap aggregating, instead of using the original data set directly, it draws n samples with replacement for m times from the original samples and uses these new m bootstrap samples as the training data set. As for parts of decision trees, random forests algorithm consists of multiple decision trees and makes some changes in the traditional decision trees. In the classical decision tree technique, a train data set first goes to the root node of a decision tree. Each decision note further split the data into two parts according to a specific predictor in the data. Until data could not be segregated further at the nodes where are named as leaf nodes, a classification result comes out. The significant difference between standard decision trees and decision trees in random forests is that traditionally each node is split based on the best split among all predictors but in random forests, each node is split with the best among a subset of variables randomly chosen from the pool of predictors. Compared with the original decision tree algorithm, random forests classification reduces variance of outcomes and solves the overfitting problem caused in decision trees.

In this report, we use if the words are uppercase or lower case and the length of words to identify names in a corpus. Based on these features, we conduct the random forests algorithm to make classifications and tune the parameters.

4.2.3 Conditional Random Fields

Conditional Random Fields(CRF) is a supervised machine learning algorithm that is commonly applied to NER tasks. CRF is only suitable for classification data sets and has developed several types. The CRF model we used is the simplest type – Linear Chain CRF. The basic unit of data being imported be CRF is a sentence. In each sentence of a corpus, every word comes sequentially. For example, we would expect a name coming after prepositions like 'to' or 'for' but never expect it coming after a pronoun like 'me'. CRF takes advantage of this structural names and try to under stand the sequential relationship between name tokens and its former tokens.

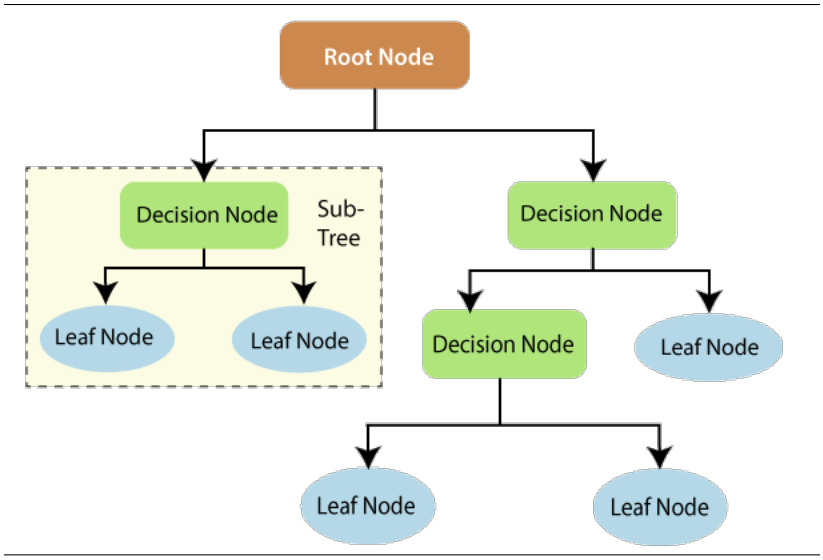

Figure 2: Structure of a decision tree

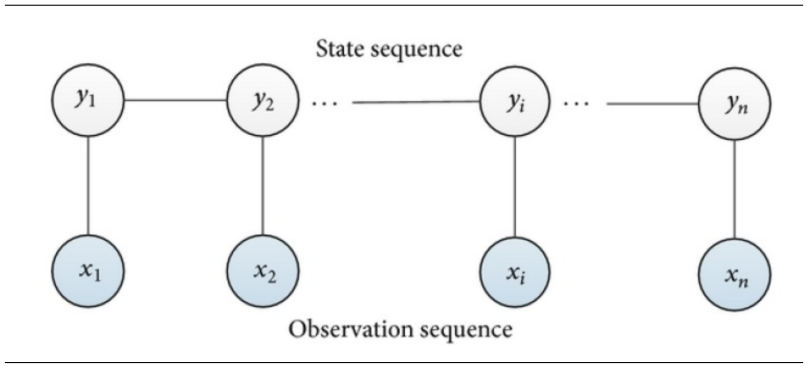

Figure 3: Structure of a CRF Linear Chain Model

In practical, Each CRF model contains its unique feature set. The feature set is divided into 2 parts: the feature set concerning the token being focused on and the feature set concerning the former or latter token of the focus token. When the focus token is the first or the last token in a sentence, the second part of the feature set is represented by 'BOS' or 'EOS' respectively. Each of the features in the feature set can be abstractly denoted as $f_j(s, i, l_i, l_{i-1})$ where s is the sentence the token in, i denote the token is the ith woken of the sentence, l_i denote the first part of the feature set and $'l_{i-1}$ denote the second part of the feature set. The actual feature set being used is complicated and can be obtained by contacting the authors.
After the feature set is determined, CRF model assign each feature with a weight λ_j. So now we can obtain the weighted score of a certain sentence.

$$score = \sum_i \sum_j \lambda_j f_j(s, i, l_i, l_{i-1})$$

When the feature set contains the tags for each token, i.e. whether a name in the NER task, The score of each sentence will depend on the tagging sequence. We denote this by putting an l in the score

function.

$$score(l) = \sum_i \sum_j \lambda_j f_j(s, i, l_i, l_{i-1})$$

By normalizing and taking exponentiation we can get the possibility of each tagging sequence.

$$p(l) = \frac{e^{score(l)}}{\sum_{l'} e^{score(l')}} = \frac{e^{\sum_i \sum_j \lambda_j f_j(s,i,l_i,l_{i-1})}}{\sum_{l'} e^{\sum_i \sum_j \lambda_j f_j(s,i,l'_i,l'_{i-1})}}$$

5 Data Analysis

5.1 Data pre-processing

In the report, we use email corpora extracted from the Enron corpus and we will use the corpora to train a model which could identify names in a cleaned document. For the above methods Random Forest and LDA, we conduct the data pre-processing for email corpora.

Since the given corpora have already been split into train and test set, we use the train set to build the model. Then in a standard email, since there are many unrelated parts such as and names in email address don't count, we only identify names in body of emails. Then we delete some symbols such as "?","!", "-" and digits in the email and also remove stop words with 'nltk' packages in python. Also we dropped all the repetitions. For random forests, we use words in original cases but in LDA, we transform all the words into lower cases for better performances in prediction.

5.2 Results and Observations

5.2.1 Random Forests

We use train set to train the parameters in the model and use test part to evaluate the performances of the model. In Figure 4, "Normal" represents words except names and it shows in prediction of 'Normal' words, it causes few mistakes. In terms of predicting names, this model has precision of 96.9% and recall of 57.3%

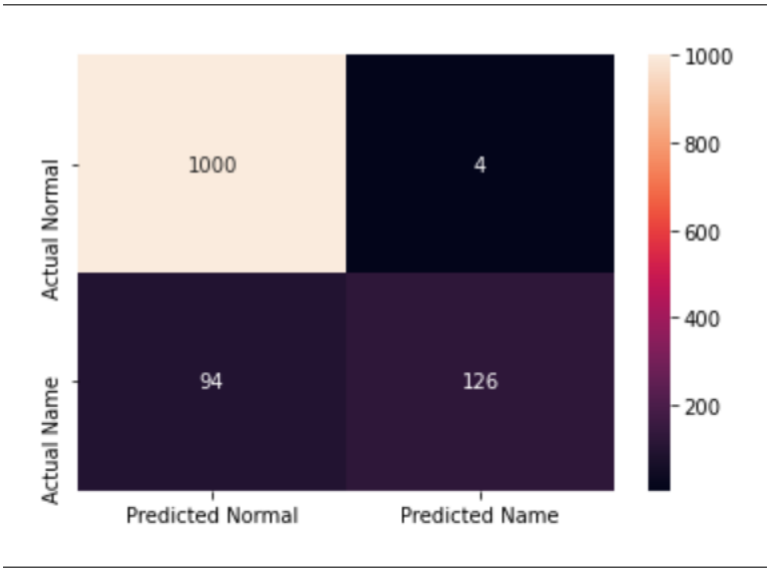

Figure 4: Confusion matrix from results of LDA

5.2.2 Latent Dirichlet Allocation

In this report, the number of latent topics is tuned as 4 topics and α and η are set as 'asymmetric' and 0.91 respectively where "asymmetric" is an available choice for α parameter in python. And similarly, we use train data to build the model and evaluate the performances of the model with test data. For this result, we denote topic 0 as name topic. And Table 1 shows that in topic 0, there are around 124 names identified and 96 names unidentified. However, topic 0 also contains many "Normal" words which are confused with names. Also, we could denote topic 0 and 1 as name topic but this will also result in misclassifying more "Normal" words as names.

Table 1: Classification results from LDA

	Train		Test	
Topic	Tag	Number	Tag	Number
0	Name	423	Name	124
	Normal	1001	Normal	447
1	Name	127	Name	66
	Normal	831	Normal	292
2	Name	15	Name	27
	Normal	180	Normal	96
3	Name	23	Name	3
	Normal	99	Normal	13

5.2.3 Conditional Random Fields

The python package used to train CRF is sklearn_crfsuite. This package requires a specific type of input instead of pandas data frames so the data form may be different with the two models above. Overall, the CRF model achieved a precision of 81.1% and a recall of 74.4%. There are some other findings. Table 2 shows the transition feature in CRF. From the table we can know that normal words tend to follow normal words and names tend to follow names and this is quite intuitive. Table 3 shows part of nontrivial state features. They may be helpful for further researchers. Note that there are some other trivial state features that even have higher importance. Most of these features are specific names. This means the CRF model remembers some specific names that may be caused by overfitting or inadequate feature set.

Table 2: Transition Features in CRF

former->latter	importance
normal->normal	2.058
normal->name	-2.379
name->normal	-1.819
name->name	0.883

6 Conclusions and discussions

Among all three models, random forest has high precision and low recall, CRF has moderate precision and recall, and LDA has the worst performance. CRF model reveals some sequential patterns of predicting name tokens. The reason why random forest models have such performance may be that there are more normal tokens than name tokens in the corpus. Thus making the data set unbalanced and harder to get a high recall.
Both random forest and crf show signs of overfitting, this suggests that the feature set being used may

Table 3: State Features in CRF

label	importance	rule
name	4.868	The former word has lower case 'mr'
name	2.431	The word ends with '-cks'
name	1.995	The word ends with '-son'
normal	2.203	The former word has lower case 'in'
normal	1.939	The word starts a sentence
normal	1.931	The word has nltk tag cc(coordinating conjunction)

still not be big enough. Nevertheless, we finally obtained a acceptable model in identifying names in email corpus and validated the efficiency of CRF models on such text scenario.

References

[1] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *the Journal of machine Learning research*, 3, 993-1022.

[2]. Liaw, A. & Wiener, M. (2002). Classification and regression by randomForest. *R news*, **2**(3), 18-22.

[3].Mbaabu.O (2020, December 11) Introduction to Random Forest in Machine Learning.*Section*. https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/

[4].Deng.H (2018, December 8) An Introduction to Random Forest.*towards data science*. https://towardsdatascience.com/random-forest-3a55c3aca46d

[5]. Azar, A.T., Elshazly, H.I., Hassanien, A.E., & Elkorany, A.M. (2014). A random forest classifier for lymph diseases.*Computer methods and programs in biomedicine*, **113**(2), 465-473.