Forwardlane Chat Agent

Urjit Patel
Center For Data Science
New York University
(up276@nyu.edu)

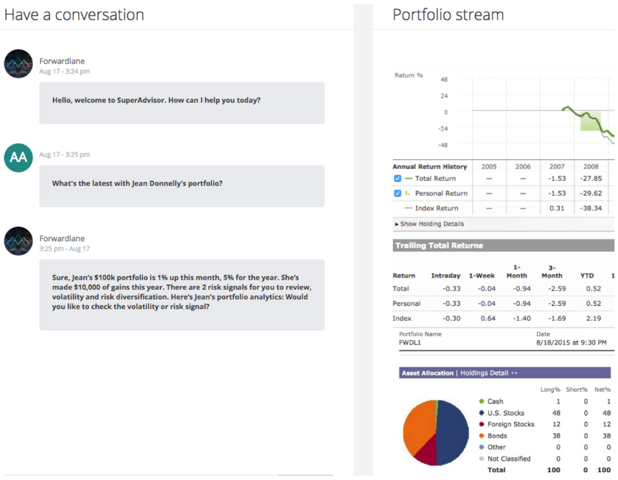
***Abstract*—In this paper we demonstrate our work done on Artificial Chat Agent which can understand Natural Financial Language. We will show how we used the natural language data for enhancement of contextual, conversational Natural Language Interface using clustering techniques on raw data, Series of Machine Learning/Deep Learning Natural language classifiers and Big data Similarity Algorithms.**

I. Introduction

With the growing Financial Data and Importance of finance domain, we see a potential role of Data Science in this specific domain. Everyday Tera Bytes of finance related data is getting generated and with that it has become necessary to have a robust Intelligent system to handle and understand these data, which can be built using Data Science techniques. On the other hand, with these much of data available now it has become possible to develop and train Intelligent systems which can understand Natural language associated with the Financial Domain.

Our aim was to enhance the Contextual Chat Interface for the user. We scraped the Financial Questions data from various web resources and also used Yahoo Question Answer data set for our experiments. With these questions set we did our experiments on Clustering raw data, on building Natural Language Classifiers using Machine Learning techniques and Big Data Similarity Algorithms.

II. Business Problem



Have a conversation

Forwardlane
Aug 17 - 3:24 pm

Hello, welcome to SuperAdvisor. How can I help you today?

AA
Aug 17 - 3:25 pm

What's the latest with Jean Donnelly's portfolio?

Forwardlane
3:25 pm - Aug 17

Sure, Jean's $100k portfolio is 1% up this month, 5% for the year. She's made $10,000 of gains this year. There are 2 risk signals for you to review, volatility and risk diversification. Here's Jean's portfolio analytics: Would you like to check the volatility or risk signal?

Portfolio stream

Annual Return History	2005	2006	2007	2008
☑ — Total Return	—	—	-1.53	-27.85
☑ Personal Return	—	—	-1.53	-29.62
— Index Return	—	—	0.31	-38.34

▸ Show Holding Details

Trailing Total Returns

Return	Intraday	1-Week	1-Month	3-Month	YTD	1
Total	-0.33	-0.04	-0.94	-2.59	0.52	
Personal	-0.33	-0.04	-0.94	-2.59	0.52	
Index	-0.30	0.64	-1.40	-1.69	2.19	

Portfolio Name
FWDL1

Date
8/18/2015 at 9:30 PM

Asset Allocation | Holdings Detail ▸▸

	Long%	Short%	Net%
● Cash	1	0	1
● U.S. Stocks	48	0	48
● Foreign Stocks	12	0	12
● Bonds	38	0	38
● Other	0	0	0
● Not Classified	0	0	0
Total	**100**	**0**	**100**

Fig. 1. Forwardlane Chat Agent Screenshot

Having an Artificial Chat Agent Which can interact with users can be very useful for any Financial domain company. If a company is able to provide a interface where user can just throw his/her query and get appropriate answer, It will surely increase user ease of access and will also increase user satisfaction. Just imagine your self as a user, How happy you would be if you have an AI system, where you can ask, ”Where should I invest?” and It will search through current market and on all current available policies and finds a best investment policies for you. You can even ask,” Tell me about my last month expenses”, and it will give you some nice statistics charts and numeric values for your last month transactions. How useful it would for you. This system can play a role of an online adviser.

III. Data Set Details

To get started, first requirement was to have a set of questions on which we can train a Natural Language Classifier. For that we scrapped around 50000 questions from potential web sources using web scrapping. Initially these questions had more than 69 different labels associated with them. We knew that It would be really difficult to train the classifier on 69 different categories just with 50000 questions. We performed some cluster analysis on these 69 small clusters and merged similar clusters into broad categories. Main Categories of these data set are as below:

$1 : healthcare$
$2 : insurance$
$3 : investement$
$4 : personal - finance$
$5 : retirement$
$6 : small - business$
$7 : taxes$

Next step was to associate answers to these questions. As Generating answers is not that easy task and also we can have different type of questions, such as general, Persona based or Context based. We need specific modules for each of these different type of answers. To start with, we just took out set of 100 questions and prepared answer templates for these questions for our experiment and also prepared a dynamic dictionary which contains the mapping between Questions and Answer templates.

Apart from that, we got the Yahoo questions and answers data set which contains around 4.5 Million questions from all domains. We are currently working on separating out Finance

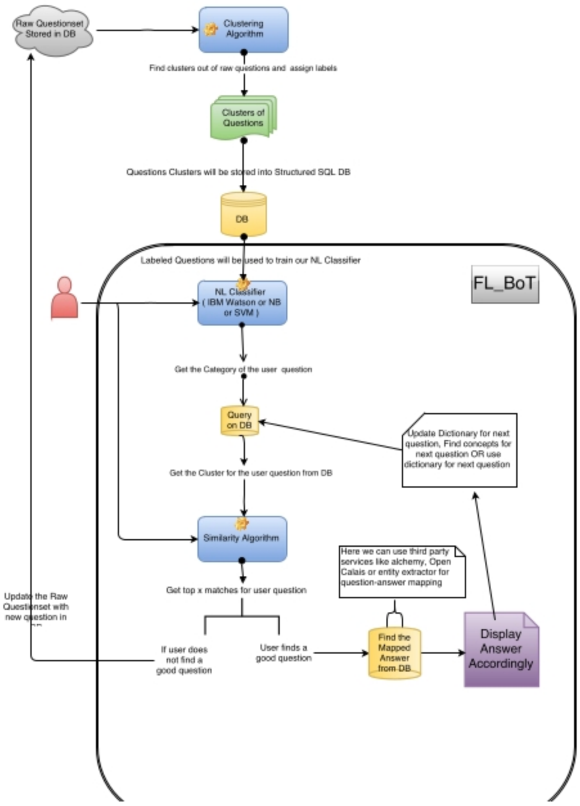

Raw Questionset Stored in DB
Clustering Algorithm
Find clusters out of raw questions and assign labels
Clusters of Questions
Questions Clusters will be stored into Structured SQL DB
DB
Labeled Questions will be used to train our NL Classifier
FL_BoT
NL Classifier (IBM Watson or NB or SVM)
Get the Category of the user question
Query on DB
Update Dictionary for next question, Find concepts for next question OR use dictionary for next question
Get the Cluster for the user question from DB
Similarity Algorithm
Here we can use third party services like alchemy, Open Calais or entity extractor for question-answer mapping
Get top x matches for user question
Update the Raw Questionset with new question in
If user does not find a good question
User finds a good question
Find the Mapped Answer from DB
Display Answer Accordingly

Fig. 2. Forwardlane Chat Agent Flow Diagram

related questions with answers. Our hope is to train a Neural Network once we got sufficient data in our hand.

Dataset descriptions:

We performed the cross validation with 80-20 ratio of train-test set from the initial 50000 questions.

Train Set - 40000 Questions $(6.54gb)$
Test Set - 10000 Questions
No of labels - 7

IV. Development Problem

Development of Chat Agent is basically divided into four sub problems.

1. Clustering of Raw Data
2. Training of Natural Language Classifier
3. Similarity Algorithm
4. Answers Fetching Module

Clustering of Raw Data:

Initially we had 50000 questions with 69 different categories. We found the important words from these small clusters using TF-IDF technique and word cloud analysis. Once we filtered out important words, we grouped all those words into one corpus and performed the TF-IDF cosine similarity between these clusters. With the help from Finance domain expertise, we were able to merge down those 69 clusters into 7 main categories.

Training of Natural Language Classifier:

Thanks to clustering analysis, now we had 50000 questions with only 7 different categories. Next and the most important step was to train Natural Language Classifier on these Questions set. Aim was to have an accurate trained classifier which can identify the category of user questions.

As part of Natural language classifiers, we mainly tried variants of two Machine Learning algorithms Naive Bayes and SVM with different optimization techniques (Batch/Mini/Stochastic Gradient Descent), hyper parameters and vectorizer techiniques (TF-IDF / Binary / Count Vectorizers). We improved our result with ensemble method using Random Forest on the top of hierarchical Natural language classifier algorithms on first layer. We are currently trying to improve the accuracy using Data Augmentation techniques and Deep Learning Convolutional Neural Networks/Recurrent Neural Networks for text classification.

What we did to improve our accuracy?

1. Removed English stop-words
2. Found the repeating words and overlapping words in all clusters and removed those words too
3. Performed the cluster analysis and used word clouds to find the relationship between these clusters
4. Merged the closely related clusters
5. Removed the Noisy data
6. Tuned the hyper-parameters
7. Used Stochastic Gradient Descent as Optimization Method
8. Used TF-IDF to create vectors TF:IDF uses the probabilistic model to assign weights according to the importance of word in each document (question)

We were able to improve the accuracy with high margin with above steps.

Similarity Algorithm:

Once we identified the category for the user question. Next task is to fetch all the questions which belongs to user question category. Once we got the cluster questions, we find top three questions similar to user's question and display it on user's screen. User can now select any of these three question to get the answer for that particular question.

To get the top match, we used TF-IDF cosine similarity score and sorted the questions into Descending order. Initially as we won't be having all the important questions, there may be some cases where user will not find the desired question into the top three matches. That is why we have added 4th option for the user which is "No match found". If uses does not find any good match then we simply take this question and put it into the separate set of new questions. Business person can now work on these type of questions and update our current question answer database.

One benefit here is that we don't need to fetch all the questions into our main memory, we just need to fetch particular cluster questions.

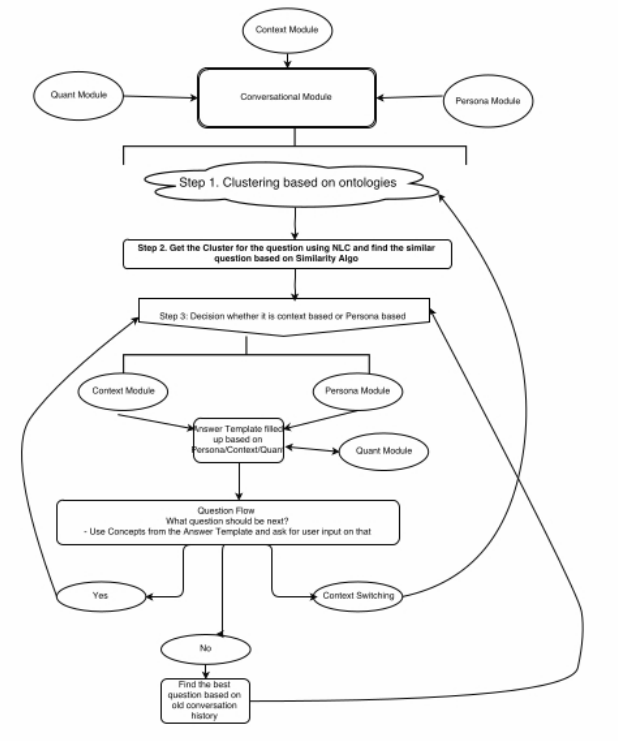

Context Module
Quant Module
Conversational Module
Persona Module
Step 1. Clustering based on ontologies
Step 2. Get the Cluster for the question using NLC and find the similar question based on Similarity Algo
Step 3: Decision whether it is context based or Persona based
Context Module
Persona Module
Answer Template filled up based on Persona/Context/Quant
Quant Module
Question Flow
What question should be next?
- Use Concepts from the Answer Template and ask for user input on that
Yes
Context Switching
No
Find the best question based on old conversation history

Fig. 3. Forwardlane Chat Agent Modules connection diagram

Answers Fetching Module:

Once user selects a question, next step is to fetch the answer for it and display on the user screen.

V. Performance Evaluation

Precision and Recall

For us, It was really very important that we don't mis-clasify user's question with wrong category. Otherwise we wound never able to find a good match questions from our database. Hence, we mainly focused on increasing precision rather than focusing on Recall.

$$p = \frac{tp}{tp + fp} \qquad r = \frac{tp}{tp + fn}$$

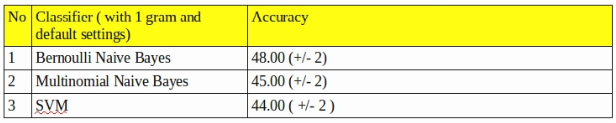
with tp being the number of true positives (i.e. points correctly predicted as positive), fp being the number of false positives, and fn being the number of false negatives.

Baseline Results:

We used Binary Vectorizer method for the baseline (simplest one) to convert questions into numeric vectors. We trained three different classifiers with defualt setting and came up with below results,

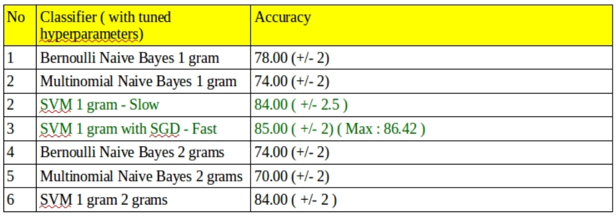
Enhanced NLC Results:

With necessary clustering, data pre processing, hyper parameter tuning and best optimization method, we improved the accuracy with high margin.



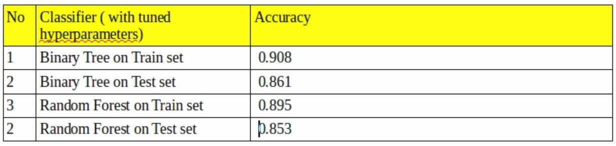
No	Classifier (with 1 gram and default settings)	Accuracy
1	Bernoulli Naive Bayes	48.00 (+/- 2)
2	Multinomial Naive Bayes	45.00 (+/- 2)
3	SVM	44.00 (+/- 2)

Fig. 4. Baseline Results



No	Classifier (with tuned hyperparameters)	Accuracy
1	Bernoulli Naive Bayes 1 gram	78.00 (+/- 2)
2	Multinomial Naive Bayes 1 gram	74.00 (+/- 2)
2	SVM 1 gram - Slow	84.00 (+/- 2.5)
3	SVM 1 gram with SGD - Fast	85.00 (+/- 2) (Max : 86.42)
4	Bernoulli Naive Bayes 2 grams	74.00 (+/- 2)
5	Multinomial Naive Bayes 2 grams	70.00 (+/- 2)
6	SVM 1 gram 2 grams	84.00 (+/- 2)

Fig. 5. Enhanced Model results



No	Classifier (with tuned hyperparameters)	Accuracy
1	Binary Tree on Train set	0.908
2	Binary Tree on Test set	0.861
3	Random Forest on Train set	0.895
2	Random Forest on Test set	0.853

Fig. 6. Ensemble Model results

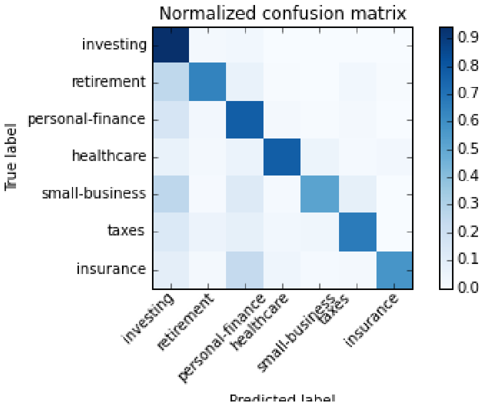


Fig. 7. Forwardlane Chat Agent Modules connection diagram

Results with Random Forest ensemble:

We combined the results achieved from variants of Naive Bayes and SVM, and applied Binary Tree and Random forest to make sure that we don't overfit. We got below results,

Confusion Matrix

Below confusion matrix shows that, we were able to get high precision for all 7 categories.

References

[1] Text Understanding from Scratch *Paper by Yann Lecun and Xiang Zhang*.

[2] http://www.wildml.com/2016/04/deep-learning-for-chatbots-part-1-introduction

[3] https://www.quora.com/Natural-Language-Processing-What-are-the-best-algorithms-p

[4] http://stackoverflow.com/questions/8897593/similarity-between-two-text-documents

[5] http://stackoverflow.com/questions/16205020/measuring-semantic-similarity-between-two-phrases