Data expansion for semantic classification

Anna Liednikova
Supervisor: Claire Gardent

February 3, 2019

Contents

1 Introduction

This project was set within the framework of a collaboration with the ALIAE startup where the aim is to develop a chatbot to collect information from clinical patients. The main idea is to replace strictly defined surveys by a more natural conversation in order to let users express themselves freely so that more information could be collected.

The chatbot consists of two main modules (cf. Figure 1):

- NLU (Natural Language Understanding): interpreting the user input
- Dialog Managment: deciding how to respond to the user input and generating the system response

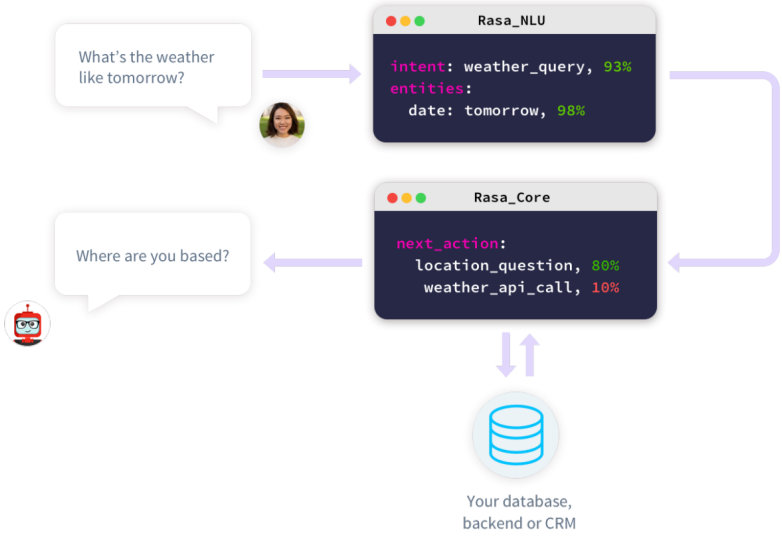


Figure 1:

In this project, we focus on natural language understanding and consider a simplified notion of understanding which maps a user input to an "intent". Given a pre-defined and finite set of intents (Does the user speaks about pain, physical activity etc. ?), the aim of the NLU module is to assign each

user input an intent[1]. The detected intent is then used, in conjunction with additional information extracted from the previous dialog interactions, to determine how to respond. Figure 1 illustrates this[2]

A small manually annotated dataset (roughly 100 sentences for each category) was available. But that's not enough for learning a good NLU model and meet the variety of users possible inputs. Annotating sentence takes plenty of time, so some automated or semi-automated way should be chosen. So this project focuses on exploring automatic ways of extending the training data and learning NLU models from these extended data.

First, we do the literature review of approaches applied to similar problems. Later in the methodology part, we will cover four parts: building sentence representation model, evaluate it by clusterisation, train a classifier for labelling the data and application semi-supervised learning for extending the dataset. Next two our datasets and preprocessing routine will be described. Experiment part contains a quantitative and qualitative description of the models and final results. In the end, we will sum up the work have been done and give some assumption for future improvements.

2 Literature review

[FLWH18] MODIFIED: present an approach for automatically generating additional training data for event extraction systems. First, they trained a baseline classifier on available data. Then, they cluster external data to obtain cluster of paraphrases using the NewsSpike method introduced by [ZHDCZ15]. They then label the clusters using the baseline model trained on the initial dataset of labelled data. Combining the new labelled data and the original one, they then retrained the event extractor.

3 Methodology

We follow [FLWH18] methodology. Instead of using [ZHDCZ15]'s methods for identifying clusters of related sentences however, we explore different ways of representing sentences using deep learning approaches. We then

[1]We make the simplifying (and incorrect) assuption that the user input contains a single intent.

[2]The figure actually shows a more complex notion of understanding where the user utterance is mapped not only to an intent but also to a set of entities. In this work, we focus on how to improve intent detection and leave entity detection for future research.

apply clustering to the resulting sentence representations. Finally, we investigate the impact of the extended labeled data on classification.

3.1 Building Sentence Representations

We explore two ways of building sentence representations: Word2vec and LDA. The good point of using these models is that they can be used for both word and sentence representation since one word can be treated as a small sentence. That can help us also to create sentence representations in two steps using machine and deep learning techniques. First, we map words to continuous representations. Second, we combine these word representations into sentence representations using BiLSTM encoder. ADD: bibref.

LDA. One of the most common methods for constructing thematic models is the Latent Dirichlet Allocation (LDA) [DMB03] that models a document as a distribution of topics and a topic as a distribution of words. In our case, we work just with sentences, so this notation will be used further.

Here is example to topics.

ADD: an example of topics produced by an LDA model

```
[(0.13819554, 'sleep'),
  (0.10817484, 'morning'),
  (0.10187448, 'sometimes'),
  (0.09861754, 'help'),
  (0.060275868, 'able'),
  (0.060111083, 'nap'),
...],
```

And every sentence described by distribution vector over the topics in the following manner:

ADD: an example of document representation as a distribution over topics

If we want represent just one word we can just treat it like a small sentence and have the same output.

This algorithm is based on the Dirichlet prior distribution and uses in its work the bag-of-words model - a model for analyzing texts that takes into account only the frequency of words, but not their order. This model is well suited for thematic modeling, since it allows you to detect implicit relationships between words. The LDA method performs soft clustering and assumes that each word in the sentence is generated by some latent topic,

which is determined by a probability distribution on the set of all words of the text.

Word2Vec. This technology works in the following manner: word2vec takes a large text corpus as input and matches each vector with a word, producing the coordinates of the output words. First, he creates a dictionary, learning on the input text data, and then calculates a vector representation of the words. A vector representation is based on contextual proximity: words that appear in the text next to identical words (and therefore, have a similar meaning) will have close coordinates of word vectors in a vector representation. The resulting word vectors can be used for natural language processing and machine learning.

$https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf$

ADD: To build sentence representations, we simply average the Word2Vec vectors of the content words present in that sentence.

Bi-LSTM. The general idea of BiLSTM layer is repeating the first recurrent layer in the network so that they are next to each other and then providing the straigh input sequence as input to the first level and an inverse copy of it for the second.

In more formal way, for the sequence of words $T\{w_t\}_{t=1,...,T}$ bidirectional LSTM computes a set of T vectors $\{h_t\}_t$. For $t[1,.,,,T]$, h_t is the combination of forward and backward LSTM, that read sentences in two opposite directions.

We experiment with two ways of combining a changing number $(h_1, ..., h_T)$ to form a vector of a fixed size, either by choosing the last hidden state h_T (BiLSTM-last), or either by selecting the maximum value over each dimension of the hidden units (max pooling) (Collobert and Weston, 2008) or by considering the average of the representations (mean pooling). The choice of these models is motivated by their demonstrated effectiveness in embedding models (Conneau et al., 2018)

3.2 Clustering Sentences

Using the sentence representations described in the previous section, we apply clustering to group together sentences that are similar. We compare three clustering algorithms:

- K-Means (KM)

- Agglomerative Clustering (AG)
- Gaussian Mixture (GM)

We set the number of clusters to the number of intents (20) which allows using not only purity and Silhouette coefficients to evaluate the clusters but also homogeneity and completeness to get the idea how resulting groups correlate with initial intents. Also, for each model, we plot a confusion matrix with a background gradient to visualize how well clusters separate different intents.

K-Means. The basic idea is to define K centroides one for each cluster. It is best to place them as far apart as possible. The next step will be to accept each point belonging to this data set and associate it with the precision of the centroid. If no point is under consideration, the first stage is completed. At the moment, we need to re-calculate K's new centroids, as the cluster barycenter that stems from the previous step. After these new centroids appear on us, a new binding must be made between the same points of the data set and the nearest new center of gravity. A loop is created. As a result of this cycle, we can see that k centroids change their location step by step until any changes are made. In other words, centroids are no longer moving.

The algorithm is aimed at minimizing the target function, in this case, the quadratic error function. Target function:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} (\|x_i^{(j)} - c_j\|)^2 \tag{1}$$

where $(\|x_i^{(j)} - c_j\|)^2$ is the chosen distance between the data point $x_i^{(j)}$ and center c_j, it is an indicator of the distance n of the data points from their respective cluster centers.

Centroid is defined as follows:

$$c_j = \frac{1}{N} \sum_{i=1}^{N} x_i^{(j)} \tag{2}$$

where $x_i^{(j)}$ is the point belonging to the cluster J, N - total number of points in the cluster.

Agglomerative Clustering. is a kind of hierarchical clustering with 'bottom-up' approach when new clusters are created by combining smaller clusters and, thus, a tree is created from leaves to trunk. The main feature of the agglomerative hierarchical clustering method is that if we want to get a different number of clusters, we will not need to restart the algorithm, because the whole tree is calculated, and then we say that we want to cut it in some place.

Different linkage criterions can be used to calculate distance between sets of observation. Wards method minimizes the variance of the clusters being merged. This method is used for tasks with closely spaced clusters. The euclidian distance between clusters is taken as the increment of the sum of squares of the distances of objects to the center of the cluster, obtained as a result of their combination:

$$\Delta = \sum_i (x_i - \bar{x})^2 - \sum_{x_i \in A} (x_i - \bar{a})^2 - \sum_{x_i \in B} (x_i - \bar{b})^2 \tag{3}$$

At each step of the algorithm, these two clusters are combined, which lead to a minimal increase in variance.

$Ward, Joe H.(1963)."Hierarchical Groupingto Optimizean Objective Function".Journal of the Ame$
$236244.doi : 10.2307/2282967.JSTOR2282967.MR0148188.$

Gaussian Mixture. REWRITE: In statistics, a mixture model is a probabilistic model for representing the presence of subpopulations within an overall population, without requiring that an observed data set should identify the sub-population to which an individual observation belongs. Formally a mixture model corresponds to the mixture distribution that represents the probability distribution of observations in the overall population without sub-population identity information.

A Gaussian mixture model represents a weighted sum of M Gaussian densities elements for D-dimensional continuous-valued data vectors:

$$p(x|\lambda) = \sum_{i=1}^{M} w_i g(x|\mu_i, \sum_i) \tag{4}$$

where each Gaussian densities element defined as

$$g(x|\mu_i, \sum_i) = \frac{1}{(2\pi)^{\frac{D}{2}} |\sum_i|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_i)' \sum_i (x-\mu_i)} \tag{5}$$

with mean vector μ_i and covariance matrix $\sum_i$. The mixture weights satisfy the constraint that $\sum_{i=1}^{M} w_i = 1$.

GMM parameters are estimated from training data using the iterative Expectation-Maximization (EM) algorithm or Maximum A Posteriori (MAP) estimation from a well-trained prior model.

$https://pdfs.semanticscholar.org/734b/07b53c23f74a3b004d7fe341ae4fce462fc6.pdf$

3.3 Classifying Sentences

After choosing the best model for representing sentences, we train classifier to label data with intents. The most common approach in similar task is to use SVC. Due to the small size for dataset it's essensial to create balanced train and test ones.

SVC (Support Vector Classifier). The main idea of the method is transfering the initial vectors into a space of higher dimension and searching for a separating hyperplane with the maximum gap in this space. Two parallel hyperplanes are built on both sides of the hyperplane separating the classes. The separating hyperplane will be the one that maximizes the distance to two parallel hyperplanes. The algorithm works under the assumption that the greater the difference or the distance between these parallel hyperplanes, the less will be the average classifier error.

3.4 Semi-supervised approach

- Inspired from [FLWH18]
- We have a classifier trained on labelled data (provided by ALIAE)
- Apply our sentence representation model to obtain features for each sentence in the unlabelled data
- Apply classifier to unlabelled data to label this data ;
- Clusterize unlabelled data (200 clusters)
- from classifier, sentences now have probability and label
- Eliminate sentences whose prob is below threshold
- Eliminate clusters with less than N members
- Data expansion: adding to training data n labelled items for each intent/category

For semi-supervised approach we use two datasets: labelled one that was provided by ALIAE (gold data) and external one that will be used to expanding. We split our gold dataset into train and test sets and keep the last one with the same items for stable evalution.

Initially, we apply our sentence representation model to the gold data to obtain vectors for them and train our classifier (baseline) on the gold train set.

Later we should clean our external dataset to make it look more similar to our initial one. For example, filtering out long sentences that with high probability will be ambiguous. This will help us to expand context of intent more gradually not facing our estimator with example it can be difficult to handle with.

We apply our sentence representation model to obtain features for each sentence of the extrenal data. Later we classify each item and take top N items with the highest probability score for each predicted intent. It helps us to reduce working dataset and computation time, though N should be high enough to extract neccesary number of new items.

For our subset we find the best clusterization by elbow method [BK14]. Later, having both labels for intent and cluster, we leave just that clusters that populated by items with maximum probability that allows us to cover all intent with new examples.

For each cluster we calculate number of intent representators and their overall distribution. Later we leave just that clusters where some intent is represented more than in average. Elements of clusters that are left will be labelled by the majority. Later we select N labelled items for each category and extend train data with them to update sentence representation and classification models.

We repeat this cycle until we won't be able to extend train dataset by additional samples or classifier score will stop to grow.

4 Datasets

For our task we have two dataset. The first one is labelled by human and the second one was retrieved from the web to extend the previous one. Our goal was to use web data to expand the training set and improve the classifier which assigns each sentence an intent.

4.1 Labelled Data

The initial dataset was created manually ADD: aliae covering 20 main possible users intents. Each sentence represents one intent. For example,

Text: After sleep, for 2-3 hours, I am better and then start feeling tired again

Intent: sleep

The distribution of labels is shown in Figure 2. There are in average, 3.4 words per sentence (min: 0, max: 12, std: 2.16367). The total number of sentences is 3305 and the vocabulary consists of 1882 content words (after removing stopwords).

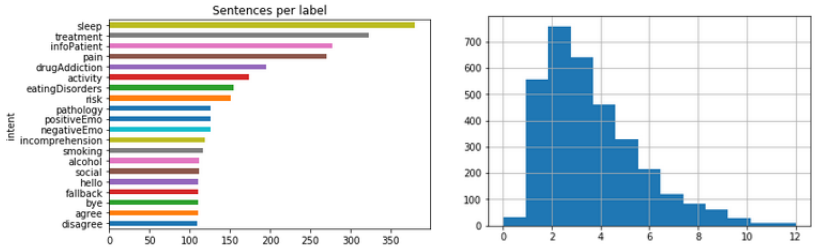


Figure 2: *Left: Label distribution in dataset, Right: Distribution of sentence length*

4.2 Forum data

We scraped 272,553 unique posts contained in each category.

To improve initial classifier we needed to extend dataset by much more data and more naturally constracted one. Looking for open medical dataset that can be used for this purpose we found [ZHDCZ15] and [SGZH10] works that used forum data for similar tasks.

HealthBoards **??** is a medical forum web portal that allows patients to discuss their ailments. We scraped 272553 unique posts contained in each of 238 categories. Figure 3 show the dataset statistics. In average sentence contains 11 ± 7 words after preprocessing.

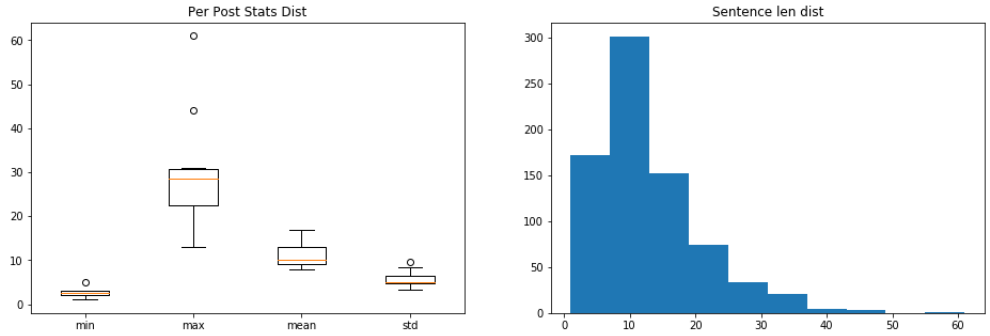


Figure 3: Text stat

This dataset has precise categorization so that some posts are more specialized and some are out of the scope of our intent labels. We should start with the data that is most similar to the initial one, so that's why we choose categories of boards that are similar to our intents. This will help us to be able gradually expand the existing context. So we will keep posts of categories provided in table 1.

ADD: scrape posts; select categories that are relevant (listed in Table 1); keep only these post that have the relevant categories + length constraints

Category of board	# of posts
digestive-disorders	5064
addiction-recovery	3644
sleep-disorders	1748
smoking-cessation	937
eating-disorder-recovery	762
chronic-pain	735
chronic-fatigue	662
stress	415
family-friends-addicts-alcoholics	312
pain-management	25

Table 1: Selected categories

Finally, the corpus consists of N sentences.

Also, for big number of iterations data should be divided in subsets by increasing sentence length because of the difference in mean values for both datasets. So after tokenizing posts into sentences we calculate it's length.

For the each iteration we should leave sentences with mean+std words in cleaned text.

5 Experiment

Experiment consists of the main stages:

- preprocess text of labelled and unlabelled data
- use clustering to choose between different ways of representing sentences
- training a classifier on the labelled data
- iterative semi-supervised learning:
  - data expansion (cf Section 3.4) ;
  - update sentence representations (using new data and old parameters)
  - retrain on classifier on gold + new labelled data
  - evaluate new classifier on gold test data

5.1 Preprocessing routine

The textual data was segmented into sentences, tokenized and lemmatized using NLP libraries. Stop words were removed.

We compared two libraries, NLTK and SpaCy.

For example, in word tokenization they give different results that can influence not only simple statistics but meaning too. Some example of different tokenization can be seen in table 2. Though concating words to ones like 'flulike' or '35mg' or 'longterm' sometimes gives more robust and concrete meaning in case of big dataset, in our case it seems better to stay with SpaCy way of tokenization in order to have smaller and more simple vocabualary.

NLTK	SpaCy
['i', 'wouldnt', 'go', 'to', 'sleep', 'until', 'like', '5', '6', 'or', '8am']	['i', 'would', 'nt', 'go', 'to', 'sleep', 'until', 'like', '5', '6', 'or', '8', 'am']
['that', 'is', 'totally', 'wrongheaded']	['that', 'is', 'totally', 'wrong', 'headed']
['i', 'am', 'in', 'the', 'process', 'of', 'tapering', 'from', 'suboxone', 'longterm', 'use']	['i', 'am', 'in', 'the', 'process', 'of', 'tapering', 'from', 'suboxone', 'long', 'term', 'use']
['i', 'had', 'an', 'onandoff', 'opiateopioid', 'habit', 'from', 'about', '2010']	['i', 'had', 'an', 'on', 'and', 'off', 'opiate', 'opioid', 'habit', 'from', 'about', '2010']
['i', 'have', 'flulike', 'pathologysymptom']	['i', 'have', 'flu', 'like', 'pathologysymptom']
['i', 'have', 'exerciseinduced', 'insomnia']	['i', 'have', 'exercise', 'induced', 'insomnia']
['i', 'm', 'supposed', 'to', 'take', '6', '35mg', 'tablets', 'a', 'day', 'but', 'i', 'have', 'taken', '20', 'today']	['i', 'm', 'supposed', 'to', 'take', '6', '35', 'mg', 'tablets', 'a', 'day', 'but', 'i', 'have', 'taken', '20', 'today']

Table 2: Tokenization comparison

For stopwords removing there were three options: nltk, spacy and the longest one. The last option was rejected due to containing words like 'want', 'stop', 'successfully' etc. that can be useful for detecting basic intents like positive or negative emotion, social. Finally nltk one was selecting because of containing shorts like 'm' from 'am', 've' from 'have'. Final dictionary contained 1882 words. Also all numbers were changed to num. Chart (4) looks fine.

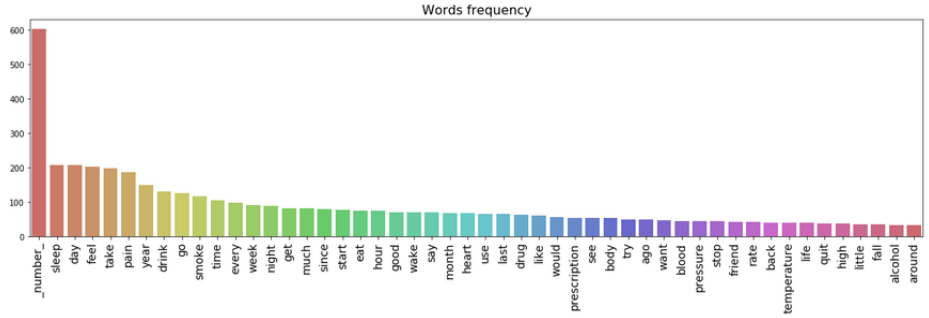


Figure 4: Words frequency

Next problem we faced is empty sentences. But there is not a lot of

occurances, so their dropping don't change the dataset much.

intent	items
fallback	12
disagree	11
hello	4
agree	3
incomprehension	2
positiveEmo	1

These sentences are: 'what?', 'same that again', 'I am up', 'same', 'can you', 'do that', 'do this', 'Will do', 'no', 'no i will not', "No i don't", 'no', 'no i will not', "No i don't", "What's up?", "What's up?", 'he', 'a', 'd', 'i', 'm', 'o', 's', 't', 'y', 'I will', 'Will do', 'No', 'no', 'No it is not', "No I don't", 'No', "I'm here"

5.2 Clustering for sentence representation evaluation

ADD: used to compare different ways of representing sentences

Forum data is splitted into 20 (number of intents) cluster to be able to see how current representation can reflect desirable labels. Also, it can provide us with information which classes will be easier to identify and which can mix with others.

For selecting the best model we changed space dimension (topics for LDA and features for W2W). Preferred number of features for W2V model is 200. It's the start point for our model.

5.2.1 Evaluation metrics

For every model following metrics and their average were calculated: purity score, Silhouette Coefficient, homogeneity and completeness scores. Later for each parameter average among clustering models calculated for each score in order to get both table and plot.

Also for each model confusion matrix created between cluster labels and initial intents.

Purity is calculated by assigning each cluster to the class that is most often found in the cluster, and then the accuracy of this assignment is measured by counting the number of correctly assigned documents and dividing by N.

$$purity(\Omega, C) = \frac{1}{N} \sum_{k} \max_{j} |\omega_k \cap c_j| \tag{6}$$

where $\Omega = \{\omega_1, \omega_2, ..., \omega_K\}$ is the set of clusters and $C = \{c_1, c_2, ..., c_J\}$ is the set of classes.

Silhouette Coefficient shows how much the average distance to objects of its cluster differs from the average distance to objects of other clusters. This value is in the range $\lfloor 0, 1 \rfloor$. Values close to -1 correspond to poor (scattered) clustering, values close to zero indicate that the clusters intersect and overlap each other, values close to 1 correspond to "dense" clearly selected clusters. Thus, the larger the silhouette, the more clearly highlighted the clusters, and they are compact, tightly grouped clouds of points.

Homogeneity and completeness. Homogeneity will be maximal if the cluster consists only of objects of one class. Completeness will be maximum if all objects from the class belong to only one specific cluster. Formally, these measures are also determined using the functions of entropy and conditional entropy, considering the partitioning of the sample as discrete distributions:

$$h = 1 - \frac{H(C|K)}{H(C)}, c = 1 - \frac{H(K|C)}{H(K)} \tag{7}$$

here is K the result of clustering, is C the true partitioning of the sample into classes. Thus, h measures how much each cluster consists of objects of one class, and c measues how much objects of one class belong to one cluster.

5.2.2 Word2Vec model

We compare results for different embedding size ranging from 10 to 600.

Embedding Dim.	purity	silhouette	homogeneity	completeness
10	0.183157	0.0680834	0.108901	0.12661
20	0.182854	0.0506986	0.111099	0.124805
30	**0.185477**	0.0683824	0.110799	0.132563
50	0.184569	0.0647393	**0.111876**	0.131964
100	0.182148	**0.0813837**	0.108022	0.138329
150	0.183661	0.0298524	0.106401	0.141179
200	0.176803	0.0581627	0.103708	0.141973
300	0.176097	0.0432732	0.106379	0.151118
400	0.169642	0.0626822	0.100455	0.148768
500	0.168533	0.0653733	0.0976706	0.146799
550	0.17176	0.0584935	0.0994422	**0.151185**
600	0.169138	0.0574187	0.0980455	0.150695

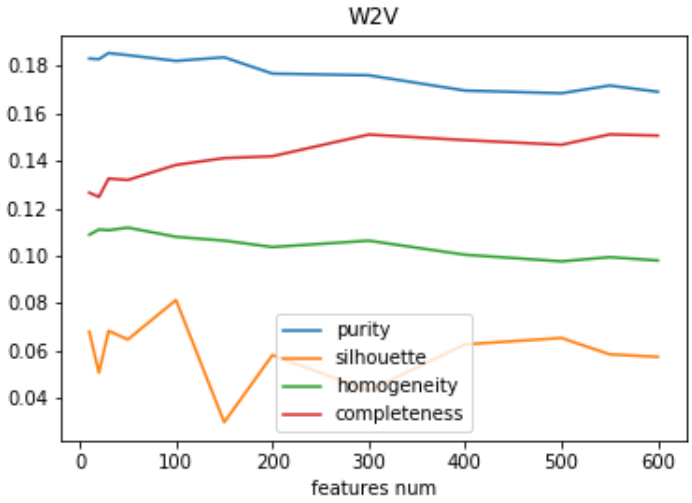
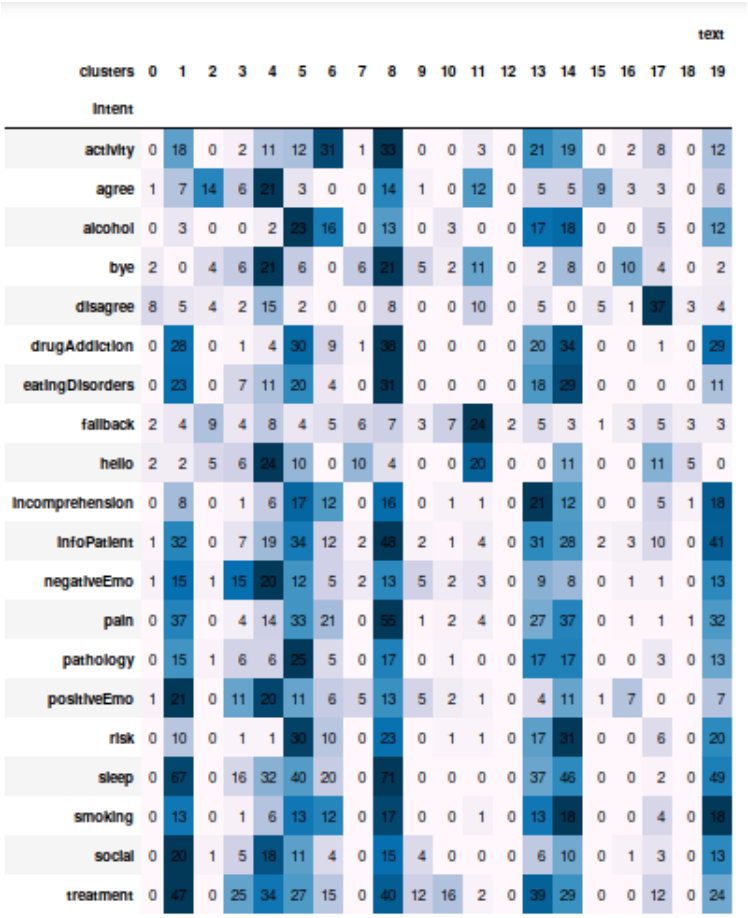


Figure 5: LDA scores

Later we will try both 10 and 100 features.



	text																			
clusters	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
intent																				
activity	0	18	0	2	11	12	31	1	33	0	0	3	0	21	19	0	2	8	0	12
agree	1	7	14	6	21	3	0	0	14	1	0	12	0	5	5	9	3	3	0	6
alcohol	0	3	0	0	2	23	16	0	13	0	3	0	0	17	18	0	0	5	0	12
bye	2	0	4	6	21	6	0	6	21	5	2	11	0	2	8	0	10	4	0	2
disagree	8	5	4	2	15	2	0	0	8	0	0	10	0	5	0	5	1	37	3	4
drugAddiction	0	28	0	1	4	30	9	1	38	0	0	0	0	20	34	0	0	1	0	29
eatingDisorders	0	23	0	7	11	20	4	0	31	0	0	0	0	18	29	0	0	0	0	11
fallback	2	4	9	4	8	4	5	6	7	3	7	24	2	5	3	1	3	5	3	3
hello	2	2	5	6	24	10	0	10	4	0	0	20	0	0	11	0	0	11	5	0
incomprehension	0	8	0	1	6	17	12	0	16	0	1	1	0	21	12	0	0	5	1	18
infoPatient	1	32	0	7	19	34	12	2	48	2	1	4	0	31	28	2	3	10	0	41
negativeEmo	1	15	1	15	20	12	5	2	13	5	2	3	0	9	8	0	1	1	0	13
pain	0	37	0	4	14	33	21	0	55	1	2	4	0	27	37	0	1	1	1	32
pathology	0	15	1	6	6	25	5	0	17	0	1	0	0	17	17	0	0	3	0	13
positiveEmo	1	21	0	11	20	11	6	5	13	5	2	1	0	4	11	1	7	0	0	7
risk	0	10	0	1	1	30	10	0	23	0	1	1	0	17	31	0	0	6	0	20
sleep	0	67	0	16	32	40	20	0	71	0	0	0	0	37	46	0	0	2	0	49
smoking	0	13	0	1	6	13	12	0	17	0	0	1	0	13	18	0	0	4	0	18
social	0	20	1	5	18	11	4	0	15	4	0	0	0	6	10	0	1	3	0	13
treatment	0	47	0	25	34	27	15	0	40	12	16	2	0	39	29	0	0	12	0	24

Figure 6: W2V confusion matrix

5.2.3 LDA model

We iterated through different numbers of topics for LDA model to see which one should be used to represent our dataset. In table 3 we can see the comparison.

num	purity	silhouette	homogeneity	completeness
10	0.253656	0.461589	0.157498	0.178928
20	0.259203	0.378276	0.168191	0.182203
30	0.251538	0.331069	0.157755	0.168448
50	0.240545	0.287032	0.147430	0.166824
100	0.266465	0.187596	0.168647	0.189221
150	0.256077	0.167888	0.161589	0.185054
200	0.263843	0.149212	0.175117	**0.194913**
300	0.250025	0.151526	0.165607	0.184314
400	**0.272718**	0.192777	**0.177547**	0.192270
500	0.259506	**0.203725**	0.174925	0.192831
550	0.229450	0.166200	0.149270	0.158548
600	0.254261	0.191729	0.171744	0.186001

Table 3: Comparision of LDA with different number of topics

The best model - is LDA with 400 topics according to scores and with priority to purity and homogeneity ones.

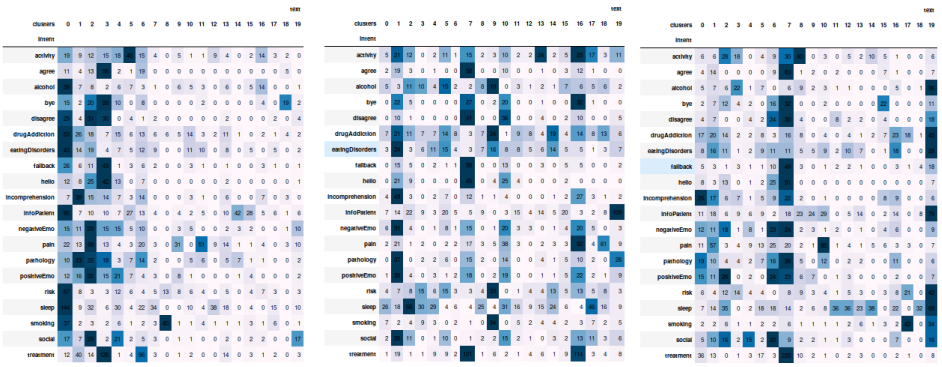


Figure 7: LDA confusion matrixes: agglomerative clustering, gaussian mixture, Kmeans

Thanks to confusion matrix that is constructed based on cluster and intent labels we can make hypotheses about possible errors and difficulties and figure out what connections our model distinguishes well. Though different clustering algorithms still capture different connections between intents, there are some common errors for all of them. In particular, the following intent are incorrectly clustered together: fallback and hello; alcohol,

drugAddiction and risk; social and emotions.

5.2.4 Google News W2V pretrained

We used word2vec google-news model with 300 dimensional feature space. CV 0.418 + 0.057

5.2.5 CNN by word

From simple encoder, w2v, lda, w2v + lda

5.2.6 BiLSTM by word

Following (et al.) we trained BiLSTM encoder model verifying the best params. So we iterated through the number of layres, it's dimention and type of pooling.

layers	purity	silhouette	homogeneity	completeness
1	**0.286369**	0.239979	**0.227069**	**0.232338**
2	0.262836	0.181804	0.211068	0.219279
3	0.238284	0.266460	0.195002	0.206703
4	0.218623	**0.439161**	0.164478	0.192599

Table 4: Comparison of models with different number of layers

So it's better to stay with one layer. Now for one layer we test its dimension.

layers	purity	silhouette	homogeneity	completeness
32	0.270273	0.244798	0.214607	0.21474
64	**0.281683**	0.22728	0.231666	0.232176
128	0.271597	0.263519	**0.234111**	**0.235486**
256	0.270681	0.267634	0.229275	0.231675
512	0.277608	0.264844	0.231268	0.235075
1024	0.266707	0.260425	0.226291	0.229238
2048	0.268796	**0.278391**	0.228211	0.231041

Table 5: Dimension choosing

Finally, we chose 64 output size due to bigger difference in purity score.

Next step is chose what encoder should we use: last hiiden layer or mean-pooling or max pool on them?

last layer	purity	silhouette	homogeneity	completeness
last hidden	0.281683	0.22728	0.231666	0.232176
mean hidden	0.255399	0.170045	0.194848	0.196056
max hidden	0.272616	0.177766	0.218339	0.219109

Table 6: Choosing pooling

Finally we got BiLSTM model with one 64-dimensional layer without pooling.

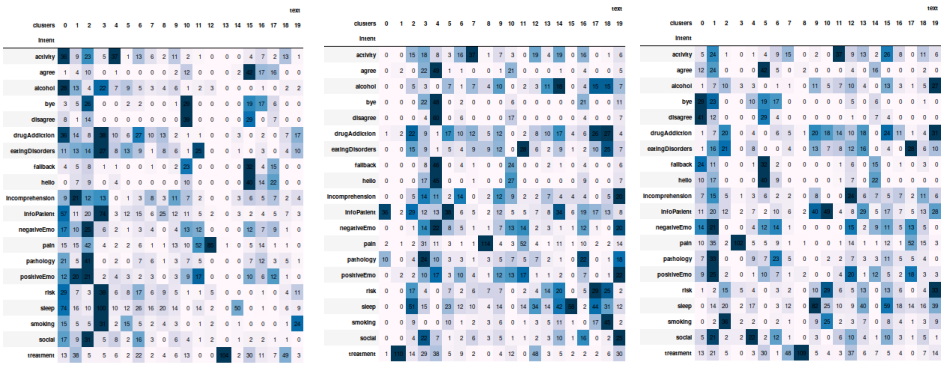


Figure 8: BiLSTM confusion matrixes: : agglomerative clustering, gaussian mixture, Kmeans

This model shows really good separatation for treatment and pain labels, tends to group together (agree, fallback and hello) and (bye and disagree).

5.2.7 Overall comparison

model	features	purity	silhouette	homogeneity	completeness
W2V	10	0.192032	0.077404	0.108761	0.127672
W2V	100	0.184266	0.065381	0.109944	0.144074
LDA	400	0.272718	0.192777	0.177547	0.192270
BiLSTM	64	**0.281683**	0.22728	0.231666	0.232176

Table 7: Overall comparison

5.3 Classifier

ADD: training on labelled data (cf. Section 4.1)

5.3.1 Evaluation metrics

A classical way to evaluate a classification model performance is using accuracy score. Sometimes, especially in case of unbalanced dataset, it can be misleading because it can represent that model learns just one class. Its better to look for a precision score as a measure of exactness and a recall score as a measure of completeness.

$$P = \frac{TP}{TP + FP} \tag{8}$$

$$R = \frac{TP}{TP + FN} \tag{9}$$

where TP - True Positives, FP - False Positives, FN - False Negatives.

If we want to get an idea how our classifier deals with both classes (in simple binary case) we should use F1 score that conveys the balance between the precision and the recall.

$$F1 = \frac{2 * P * R}{P + R} \tag{10}$$

There is a more clean and unambiguous way to describe the model behaviour that is a confusion matrix. The bigger values on the main diagonal the better as well as the more balanced small values on antidiagonal.

	Real positive	Real negative
Predicted positive	True Positive	False Positive
Predicted negative	False negative	True negative

Table 8: Confusion matrix template

5.3.2 Results

For training train and test dataset were created with 65 and 43 items respectively that makes 60/40 ratio. Both are balanced ADD: each category has the same number of items.

SVC model was chosen with the best parameters by using Grid Search. We are comparing two kernels - rbf (C=10, gamma=1) and linear (C=10, gamma=0.001). For each model, we use 5 fold cross validation. 5-folds Cross Validation error on train set 0.401 + 0.0115 and on test set 0.371 + 0.0338. Small

Kernel	Train	Test	precision	recall	F1
rbf	0.524 + 0.00717	0.387 + 0.0341	0.308	0.316	0.287
linear	0.371 + 0.00787	0.338 + 0.0431	0.238	0.241	0.215

Table 9: CV score comparison

Though model with rbf model has higher scores, the linear one is less prone to overfitting as shown by the small difference between train and test errors. It was decided to try both and observe their behavior.

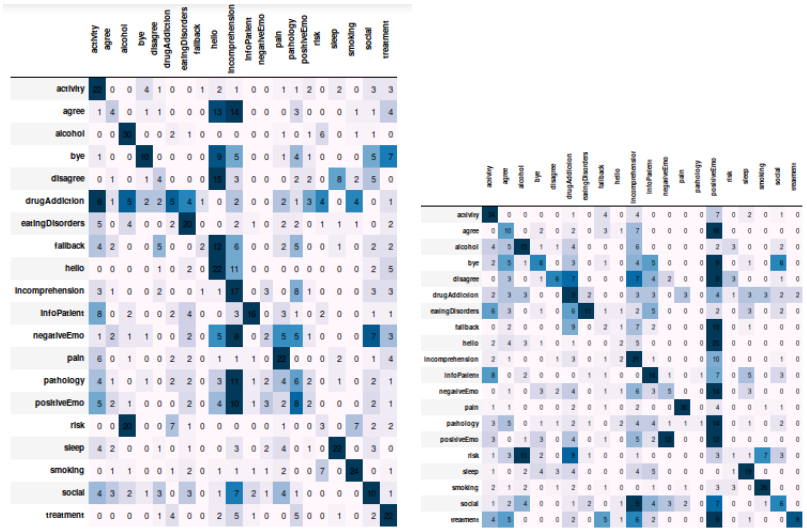


Figure 9: SVC confusion matrix

5.4 Semi Supervised Learning with linear kernel

ADD: now give classifier results obtained after each data expansion step (cf. 3.4.

Iteration 0 (Baseline). ADD: no data expansion; trained on gold data

label	precision	recall	f1-score	support
activity	0.56	0.36	0.44	66
agree	0.23	0.19	0.21	54
alcohol	0.35	0.34	0.34	44
bye	0.14	0.24	0.18	25
disagree	0.14	0.40	0.21	15
drugAddiction	0.21	0.12	0.15	75
eatingDisorders	0.26	0.61	0.36	18
fallback	0.05	0.09	0.06	22
hello	0.05	0.17	0.07	12
incomprehension	0.49	0.20	0.29	103
infoPatient	0.33	0.25	0.29	55
negativeEmo	0.12	0.21	0.15	24
pain	0.70	0.81	0.75	37
pathology	0.02	0.50	0.04	2
positiveEmo	0.28	0.07	0.11	178
risk	0.02	0.08	0.04	12
sleep	0.44	0.47	0.46	40
smoking	0.60	0.68	0.64	38
social	0.14	0.20	0.16	30
treatment	0.19	0.80	0.30	10
micro avg	0.27	0.27	0.27	860
macro avg	0.27	0.34	0.26	860
weighted avg	0.33	0.27	0.27	860

Table 10: Classifier 0

Iteration 1 For the first iteration we created subset of external data with $N = 500$ sentences for each predicted intent, then splitted data into 200 clusters, majority threshold=, hreshold for probability was chosen pretty low(0.12) in order to be able to extend all 20 categories. In result, we achived 15 new items.

label	precision recall f1-score support			
activity	0.30	0.27	0.28	49
agree	0.16	0.28	0.21	25
alcohol	0.53	0.44	0.48	52
bye	0.33	0.41	0.36	34
disagree	0.14	0.08	0.10	78
drugAddiction	0.21	0.29	0.24	31
eatingDisorders	0.37	0.73	0.49	22
fallback	0.12	0.31	0.17	16
hello	0.00	0.00	0.00	19
incomprehension	0.37	0.26	0.31	61
infoPatient	0.49	0.34	0.40	61
negativeEmo	0.05	0.06	0.05	31
pain	0.60	0.46	0.53	56
pathology	0.05	0.09	0.06	23
positiveEmo	0.37	0.11	0.17	142
risk	0.12	0.16	0.13	32
sleep	0.42	0.50	0.46	36
smoking	0.44	0.59	0.51	32
social	0.14	0.21	0.17	28
treatment	0.58	0.78	0.67	32
micro avg	0.29	0.29	0.29	860
macro avg	0.29	0.32	0.29	860
weighted avg	0.33	0.29	0.29	860

Table 11: Classifier 1

Iteration 2 For the second iteration we created subset of external data with $N = 200$ sentences for each predicted intent, then splitted data into 200 clusters, majoiry threshold=, hreshold for probability was chosen pretty

low(0.1) in order to be able to extend all 20 categories. In result, we achived 13 new items.

label	precision recall f1-score support			
activity	0.42	0.29	0.34	63
agree	0.05	0.12	0.07	17
alcohol	0.56	0.49	0.52	49
bye	0.53	0.38	0.44	61
disagree	0.37	0.38	0.38	42
drugAddiction	0.23	0.23	0.23	43
eatingDisorders	0.28	0.34	0.31	35
fallback	0.00	0.00	0.00	6
hello	0.21	0.43	0.28	21
incomprehension	0.35	0.20	0.25	76
infoPatient	0.49	0.44	0.46	48
negativeEmo	0.16	0.21	0.18	34
pain	0.70	0.81	0.75	37
pathology	0.05	0.29	0.08	7
positiveEmo	0.09	0.14	0.11	29
risk	0.07	0.11	0.08	28
sleep	0.49	0.54	0.51	39
smoking	0.49	0.66	0.56	32
social	0.26	0.31	0.28	36
treatment	0.56	0.15	0.24	157
micro avg	0.32	0.32	0.32	860
macro avg	0.32	0.32	0.30	860
weighted avg	0.40	0.32	0.33	860

Table 12: Classifier 1

Iteration	Score	CV mean	CV std	precision	recall	F1
0	0.2651	0.337	0.0503	0.265	0.340	0.262
1	0.2895	0.323	0.0212	0.289	0.319	0.290
2	0.3174	0.333	0.0118	0.317	0.325	0.304

Table 13: Classifier scores comparison

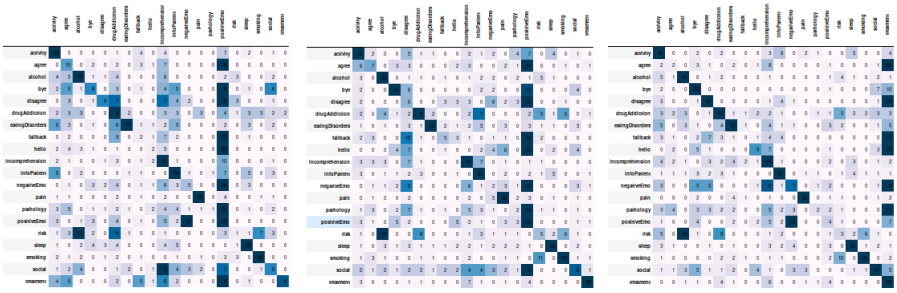


Figure 10: SVC confusion matrix

5.5 Semi-supervised learning with rbf kernel

Iteration 0.

intent	precision	recall	f1-score	support
activity	0.44	0.40	0.42	47
agree	0.72	0.20	0.32	152
alcohol	0.51	0.44	0.47	50
bye	0.35	0.29	0.32	51
disagree	0.33	0.38	0.35	37
drugAddiction	0.09	0.13	0.11	30
eatingDisorders	0.33	0.45	0.38	31
fallback	0.00	0.00	0.00	8
hello	0.23	0.25	0.24	40
incomprehension	0.51	0.29	0.37	76
infoPatient	0.30	0.54	0.39	24
negativeEmo	0.12	0.19	0.14	26
pain	0.30	0.22	0.25	59
pathology	0.09	0.16	0.12	25
positiveEmo	0.09	0.17	0.12	24
risk	0.05	0.17	0.07	12
sleep	0.42	0.58	0.49	31
smoking	0.65	0.61	0.63	46
social	0.33	0.50	0.39	28
treatment	0.51	0.35	0.42	63
micro avg	0.32	0.32	0.32	860
macro avg	0.32	0.32	0.30	860
weighted avg	0.42	0.32	0.34	860

Table 14: Intention mean probability

Iteration 1 For the first iteration we created subset of external data with $N = 500$ sentences for each predicted intent, then splitted data into 1000 clusters, majoiry threshold=, hreshold for probability was chosen pretty low(0.15) in order to be able to extend all 20 categories. In result, we achived 6 new items.

label	precision	recall	f1-score	support
activity	0.49	0.54	0.51	39
agree	0.23	0.31	0.27	32
alcohol	0.56	0.41	0.47	59
bye	0.28	0.67	0.39	18
disagree	0.33	0.31	0.32	45
drugAddiction	0.26	0.39	0.31	28
eatingDisorders	0.47	0.67	0.55	30
fallback	0.26	0.12	0.17	88
hello	0.51	0.25	0.34	88
incomprehension	0.30	0.29	0.30	45
infoPatient	0.42	0.35	0.38	52
negativeEmo	0.05	0.06	0.05	36
pain	0.65	0.49	0.56	57
pathology	0.14	0.35	0.20	17
positiveEmo	0.14	0.20	0.16	30
risk	0.07	0.14	0.09	21
sleep	0.58	0.66	0.62	38
smoking	0.51	0.54	0.52	41
social	0.19	0.29	0.23	28
treatment	0.63	0.40	0.49	68
micro avg	0.35	0.35	0.35	860
macro avg	0.35	0.37	0.35	860
weighted avg	0.40	0.35	0.36	860

Table 15: Classification report for first iteration

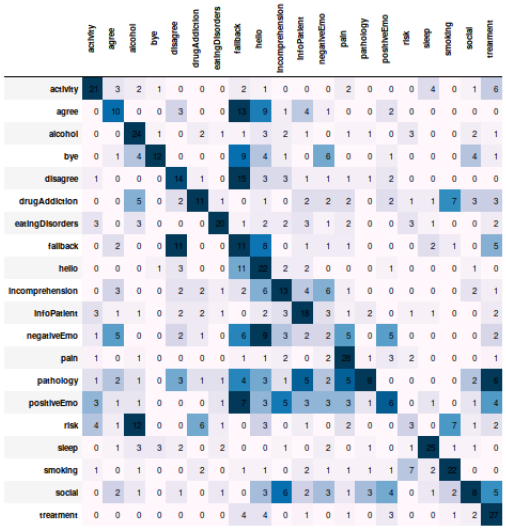


Figure 11: SVC confusion matrix

Iteration 2. For the first iteration we created subset of external data with $N = 500$ sentences for each predicted intent, then splitted data into 1000 clusters, majoiry threshold=, hreshold for probability was chosen pretty low(0.11) in order to be able to extend all 20 categories. In result, we achived 5 new items.

label	precision	recall	f1-score	support
activity	0.37	0.34	0.36	47
agree	0.14	0.21	0.17	29
alcohol	0.47	0.39	0.43	51
bye	0.33	0.30	0.31	46
disagree	0.35	0.43	0.38	35
drugAddiction	0.28	0.29	0.28	42
eatingDisorders	0.33	0.25	0.28	57
fallback	0.14	0.12	0.13	51
hello	0.40	0.19	0.26	90
incomprehension	0.42	0.43	0.42	42
infoPatient	0.37	0.52	0.43	31
negativeEmo	0.14	0.18	0.16	34
pain	0.49	0.44	0.46	48
pathology	0.16	0.29	0.21	24
positiveEmo	0.16	0.20	0.18	35
risk	0.05	0.11	0.06	19
sleep	0.56	0.63	0.59	38
smoking	0.58	0.54	0.56	46
social	0.21	0.56	0.31	16
treatment	0.40	0.22	0.28	79
micro avg	0.32	0.32	0.32	860
macro avg	0.32	0.33	0.31	860
weighted avg	0.34	0.32	0.32	860

Table 16: Classification report for the second iteration

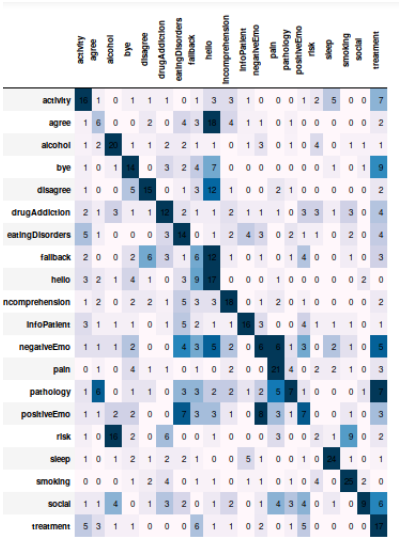


Figure 12: SVC confusion matrix

Overall comparison In this section we provide grouped results about classifier scores chaging by iteration.

Iteration	Score	CV mean	CV std	precision	recall	F1
0	0.3186	0.371	0.0338	0.308	0.316	0.287
1	0.3523	0.350	0.0153	0.352	0.371	0.346
2	0.3167	0.332	0.0150	0.316	0.331	0.313

Table 17: Classifier scores comparison

Intent	It 0	It 1	It 2
activity	0.31+0.10	0.51+0.02	
agree	0.17+0.06	0.32+0.01	
alcohol	0.37+0.13	0.53+0.03	
bye	0.17+0.03	0.40+0.03	
disagree	0.19+0.03	0.27+0.03	
drugAddiction	0.22+0.06	0.32+0.03	
eatingDisorders	0.33+0.13	0.66+0.07	
fallback	0.16+0.03	0.19+0.05	
hello	0.17+0.02	0.31+0.14	
incomprehension	0.20+0.04	0.46+0.02	
infoPatient	0.39+0.18	0.71+0.05	
negativeEmo	0.17+0.03	0.34+0.06	
pain	0.36+0.14	0.67+0.05	
pathology	0.16+0.04	0.39+0.06	
positiveEmo	0.19+0.04	0.27+0.03	
risk	0.28+0.07	0.31+0.03	
sleep	0.42+0.18	0.96+0.01	
smoking	0.49+0.17	0.79+0.02	
social	0.23+0.07	0.38+0.04	
treatment	0.25+0.04	0.43+0.04	

Table 18: Intention mean probability

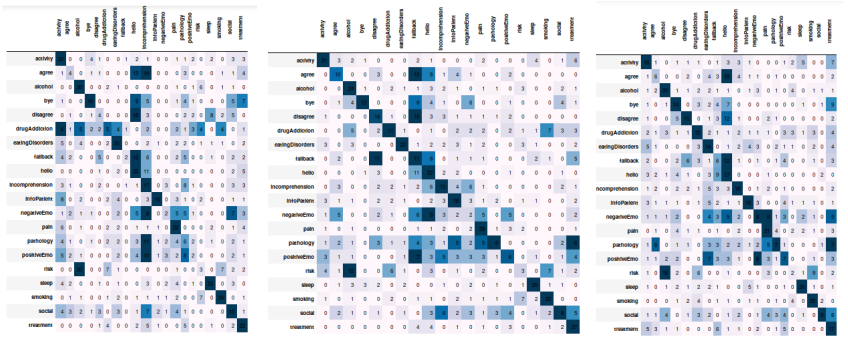


Figure 13: SVC confusion matrix

6 Conclusion and future work

In this project we were dealing with task of paraphrase detection for expanding initial dataset. Our approch is to train the word embedding model that will be able to provide clusters similar to initial classes. We used LDA and W2V models for that and later trained SVC and Kmeans models to label external data and chose the representative example to extend existing dataset and retrain models on it.

Our results shows how high rate of confusion between labels in intial classifier increases this confusion in futher data expansion though in general algorithms tends to converge to be more precise. Also, the more data you have the higher chances to find more similar sentences to initial one so that possible context for each intent will grow gradually.

Future work will be devoted to improving word embedding model, trying one-shot learning model in order to improve classification score. Transfering from one-intent to multi-intent is preferable due to the fact that sentences are usually ambiguous than not. Another possible direction for development can be creating framework for adding new labels to the existing classifier.

7 References

References

[BK14] Purnima Bholowalia and Arvind Kumar. Article: Ebk-means: A clustering technique based on elbow method and k-means in wsn. volume 105, pages 17–24, November 2014. Full text available.

[DMB03] Michael I. Jordan David M. Blei, Andrew Y. Ng. Latent dirichlet allocation. volume 3, 2003.

[FLWH18] James Ferguson, Colin Lockard, Daniel Weld, and Hannaneh Hajishirzi. Semi-supervised event extraction with paraphrase clusters. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 359–364. Association for Computational Linguistics, 2018.

[SGZH10] Parikshit Sondhi, Manish Gupta, ChengXiang Zhai, and Julia Hockenmaier. Shallow information extraction from medical fo-

rum data. In *Coling 2010: Posters*, pages 1158–1166, Beijing, China, August 2010. Coling 2010 Organizing Committee.

[ZHDCZ15] Thomas Zhang, Jason H D Cho, and Chengxiang Zhai. Understanding user intents in online health forums. *IEEE journal of biomedical and health informatics*, 19, 03 2015.

$WardJ.H.Hierarchicalgroupingtooptimizeanobjectivefunction//J.oftheAmericanStatisticalAssoci$