Using Probability Distribution over Classes in Automatically Obtained Training Corpora

Oleg Durandin[1,2], Nadejda Hilal[1,2], Dmitry Strebkov[2], Nikolai Zolotykh[1]
[1]Lobachevsky State University of Nizhny Novgorod, Nizhny Novgorod, Russia
[2]Dictum Ltd., Nizhny Novgorod, Russia
oleg.durandin@gmail.com, {hilal, strebkov}@dictum.ru,
zolotykh@vmk.unn.ru

***Abstract*—The paper contains a take on the classification problem variation featuring class noise where each object in the training set is associated with a probability distribution over the class label set instead of a particular class label. That type of task was illustrated on the complex natural language processing problem – automatic Arabic dialect classification. In the task we have a set of objects that were labeled by a heuristic rule; which could cause errors during automatic annotation process. Suggested approach allows taking into account probabilities of these errors. Described experiments show that even relatively simple accounting of that probabilities helps to significantly improve the quality of the built classifier.**

I. Introduction

It is a fact that machine learning classification problem requires big set of annotated (labeled) data. In most of the cases such annotation process is performed by human specialists and requires significant time and financial resources.

At the same time, usage of different Internet-resources could help to acquire a huge amount of data with relatively small expenses. Often it is possible to create a system of heuristic rules that will classify collected instances to a certain class with some probability. Certainly, such classification does not pretend to demonstrate 100% accuracy as it would be in case of manual classification.

Describing collected instances further, we could say that they are affected by so-called label noise effect [1,2], when an instance is associated with a corrupted label. That label differs from a correct label of the class that the instance actually belongs to.

In this paper we try to take into account this noise and provide to classifier auxiliary data in addition to information about training instances and their labels. That auxiliary data represents a measure of confidence in label associated with some instance: label's probability distribution. Suggested approach is illustrated on nontrivial task – classification of Arabic dialects. The task allows to use data mined from Twitter social network and heuristic rules that perform labeling. It is needed to mention that these rules do not work ideally, so there is the measure of confidence for each class of tweets.

II. Related works

It is known that supervised learning algorithms handle the classification problem using labeled training data, while semi-supervised learning algorithms tend to improve the performance of the classification with additional amount of unlabeled samples.

One category of learning problems is characterized by a situation when the class membership of training instance is assessed by an expert and encoded in the form of possibility distribution. At this point each example i consists of a feature vector x_i and a possible label $(u_1^i, u_2^i, ..., u_c^i)$, where u_k denotes the possibility of the fact that i-th example belongs to the class k. In [3] this problem was solved in the framework of Evidence Theory.

The same task was considered in [4] for solving the problem of parameter estimation in statistical models. The author of [4] proposed a method that is based on the maximization of generalized likelihood criterion, which can be interpreted as a degree of agreement between the statistical model and uncertain observations. This method was applied to uncertain data clustering using finite mixture models.

In our research we proposed another approach that is based on substitution of class label with corresponding probability distribution and showed that resulting algorithm improved the quality of the classification in our experiments.

III. Task of using probability distribution in training set

Assuming we have a set of *objects* X, set of *possible responses* Y, and target function $y^*: X \to Y$, each values of it $y_i = y^*(x_i)$ are known only on a finite subset of objects $\{x_1, ..., x_l\} \subset X$. Pairs «object-response» (x_i, y_i) are known as *precedents*. Aggregate of pairs $X^n = (x_i, y_i)^n$ is called a *training set* (n is the number of instances in training set).

The problem of learning from precedents could be described as follows: restore a dependency y^* from training set X^l, i.e. plot a decision function $a: X \to Y$, that approximates target function $y^*(x)$ not only on objects from training set, but on whole set of objects X [5].

If $Y = \{1, ..., M\}$, i.e. set of objects is a finite and discrete one, we have a classification on M disjoint classes. In that case

ISBN 978-952-68397-6-9 (paperback), ISBN 978-952-68397-7-6 (PDF)

whole set X is divided on classes $K_y = \{x \in X : y^*(x) = y\}$, and algorithm $a(x)$ is expected to answer the question «what class does x belong to?».

So, let's consider M-classes classification task and make the following assumption: training set is represented as a set $(x_j, (p_1, p_2, \ldots, p_m))$, where $\sum p_i = 1$, $p_i \geq 0$ and p_i is a prior probability of the fact that j-th object belongs to class i. The problem remains the same: build the decision function, $a: X \rightarrow Y$, which demonstrates optimal approximation of target function y^*.

Described task could be very actual in cases when there is no ability to annotate big set of data for training due to limitation in resources. At the same time, the task gives an opportunity to use Internet-resources, get big collection of objects and annotate them heuristically, according to some set of rules that in their turn, however, do not work perfectly.

IV. Automatic Arabic dialect classification task

Described formulation of the problem, when instances from training set are characterized by classes' probability distributions instead of concrete class labels, has been applied to such Arabic NLP problem as *automatic Arabic dialect classification*.

The term "Arabic language" actually includes different variations of single language [6,7]. It might be one of the following:

- Classical Arabic (the language of the Koran);
- Modern Standard Arabic (MSA);
- Dialectal Arabic (DA), which in its turn has different kinds based on geographical features.

Automatic classification of dialectal Arabic is aimed to identify a type of the dialect that some text or passage was written in.

It is essential to mention that most of the dialects could be characterized by some special unique and frequent dialectal words – *word-marks*. These word-marks include some pronouns, interrogative and negative particles, few nouns and verbs.

Actually, classification which is driven only by lexis would be difficult and erroneous. The reason for that is based on such peculiarities as significant similarity between different dialects and the fact that many texts that need to be processed do not contain any word-marks. Therefore, it is reasonable to use a statistical approach for that task.

Actual and detailed formulation of the problem is described in [6].

Twitter social network was used to mine required data because in most of the cases tweets contain informal text written by users. So, Twitter looks like an ideal source of texts written in different Arabic dialects. In addition to that, since some particular dialect could be characterized by its word-marks that could be met in text written only in that dialect, we could come out with the following heuristic rule: if text contains word-mark that belongs to dialect D_i, then the whole given text is written in that dialect D_i. DA texts could contain the next 3 types of words:

- MSA words;
- Dialectal words that are used in several dialects;
- Dialectal words that are unique for some particular dialect;

Collected set (see below) contains word-marks that belong to the last 2 types. Hence, dialectal text in general could contain word-marks of different dialects, and only one word-mark could help to identify a type of used dialect. Moreover, text written in DA could completely miss word-marks, and classification in that case could be based on text's style, orders of words and clitics.

V. Approach and experiments

A. Description of analyzed data

First of all, we performed mining from Twitter and collected tweets that contain word-marks. These word-marks belonged to the following 6 Arabic dialects: Saudi Arabia, Levantine, Algerian, Egyptian, Iraq and Jordan. These dialects were chosen as the most popular in Arabic countries. In addition to that, the 7-th class was added to our experiment, that class represented MSA. Tweets were grabbed in two time intervals: June 2015 (corresponding set further will be named as $Tweets_{2015}$), and April-May 2016 ($Tweets_{2016}$). Tweets from $Tweets_{2015}$ set were given to Arabic linguists for manual annotation and they returned 54,006 labeled tweets (we will refer to that set as $Test_{100}$). Figure 1 below shows a distribution of each of the classes in annotated set.

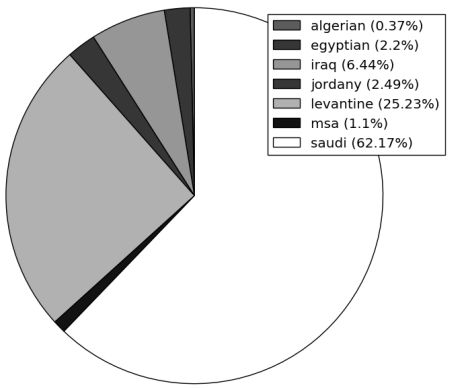

Fig. 1. Classes' distribution in annotated set

As for $Tweets_{2016}$ set, it contained 4,156,837 tweets; figure 2 describes classes' distribution that we got using heuristic rules on that set. It's needed to mention that we omitted tweets that contained word-marks belonging to two or more different dialects.

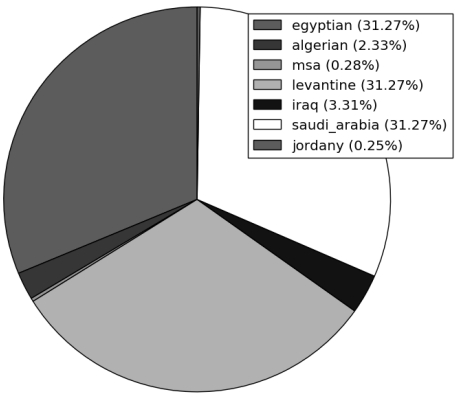

Fig. 2. Classes' distribution in automatically collected set

From the last figure we could notice that Egyptian, Levantine and Saudi Arabia dialects are distributed almost equally, while the rest of the classes contain much less corresponding data. It is caused by the fact that three dialects named above are most popular in Arabic social networks.

An imbalance between parts of different dialects in annotated set is observed because annotation activities are still in progress. Also the annotation is being performed in a consecutive way (without any randomization of the annotated set).

B. Calculation of classes' probability distributions

Firstly, let's introduce several designations. $d_{i,gold}$ will represent an event that some particular tweet truly belongs to the dialect d_i, and $d_{i,emp}$ is an event that some tweet mined automatically by heuristic rule is marked by that rule as d_i.

Therefore, $p(d_{i,gold} \mid d_{i,emp})$ is conditional probability that a tweet marked by rule as d_i will be labeled with the same dialect d_i in case of human annotation.

It is needed to mention that described probability is unknown, however, since set $Test_{100}$ is a subset of the set $Tweets_{June}$, we can estimate $p(d_{i,emp} \mid d_{i,gold})$ as a prior probability that tweet from annotated set with label d_i is contained in the set $Tweets_{2015}$ (and marked by heuristic rule as d_i accordingly).

That probability could be calculated as the following:

$$p(d_{i,emp}|d_{i,gold}) = C[d_{i,emp}=d_{i,gold}] / C[d_{i,gold}],$$

where $C[d_{i,emp} = d_{i,gold}]$ is a number of tweets labeled as d_i by both linguists and heuristic rule, and $C[d_{i,gold}]$ is a number of tweets labeled as d_i by linguists.

So, using Bayes theorem, we could calculate the original probability that is required $p(d_{i,gold} \mid d_{i,emp})$:

$$p(d_{i,gold} \mid d_{i,emp}) = p(d_{i,gold})\, p(d_{i,emp}|d_{i,gold}) / p(d_{i,emp}),$$

where $p(d_{i,gold})$ and $p(d_{i,emp})$ are measured on annotated and heuristically marked datasets accordingly .

Probability $p(d_{i,\ gold})$ represents the number of annotated instances with mark d_i to the whole amount of annotated tweets. Probability $p(d_{i,emp})$ is actually a portion of tweets in the set $Tweets_{2015}$ that belong to dialect d_i.

After all these calculations for each of the classes we come out with required probability distributions:

$$p_j = (p_{j1}, p_{j2}, \ldots, p_{jk}) \quad (1)$$

where j – instance index from training set and k is a total number of handled classes (7 in our case).

C. Description of the experiment

Let T_{train} be representation of the training set and T_{test} – representation of the test set.

For our training set we took around 1/3 of manually annotated tweets (T_{100}).

Since such set was too small for training phase, we added to the training set big collection of instances T_{emp}, which were collected automatically using heuristic rules. As these data objects are accurate only with some probability, we used probability distributions (1) for each of the classes (dialects) instead of labels.

Remaining part of T_{100} was used for testing phase.

Classification procedure was done by ensemble algorithms, in particular, *Random Forests* [8] algorithm as the most popular one.

As for features, we selected character-based n-grams (bigrams and trigrams) and vocabulary of word-marks that were used in heuristic rules to describe our instances.

At the beginning we ran Random Forests algorithm without any usage of probability distributions (so, training objects were characterized by classes that were set by heuristic rule). The algotithm showed relatively small performance on these parameters: **precision = 0.44, F_1-measure = 0.42.**

In common, such results do not look surprising since classic classification algorithms show poor results on noisy data [9].

To overcome that, we did the following: set T_{emp} was divided on non-intersected subsets based on labels, i.e.

$T_{emp} = T_{1,emp} \cup T_{2,emp} \cup \ldots \cup T_{k,emp}$, where $T_{i,emp}$ – set of instances belonging to class i. Then each of the subsets got new class label; that was done in accordance with probability distribution over classes of j-th instance $p_j = (p_{j1}, p_{j2}, \ldots, p_{jk})$.

After that procedure we got resulting set T_{prob} - set of instances that represents T_{emp} after label substitution according with the probability distribution (see above). Finally, we were able to form a training set T_{train} as $T_{train} = T_{100} \cup T_{prob}$.

Then described set T_{train} was used to train Random Forests based ensemble classifier. At the end we reached the following results of that intermediate algorithm: **precision = 0.57, F_1-measure = 0.59.**

So we got improvements for all metrics using that rather trivial idea.

D. Algorithm of classification based on probability distributions in training set

Now let's assume that training set T_{train} is an aggregate $(X^n_{train}, (p_{i1},p_{i2},...,p_{ik})^n)$, i.e. an aggregate of objects and classes' probability distributions for each object from training set (n is a number of instances in training set). Naturally, if we know correct class for some object from training set, probability of that class becomes 1.0 and probabilities of all other classes become 0.

The task of building a classifier that minimizes the error on training set looks the same: testing (and further usage) is performed on instances that have (or require) unambiguous accordance with the class.

Algorithm 1: Getting ensemble of classifiers
Input: $T_{train} = (X^n_{train}, (p_{i1},p_{i2},...,p_{ik})^n)$,
m – number of estimators
Output: Classifier

estimators_list = {}
for j = 1, ..., m:
 $vec\_classes_j$ = **generate_classes_by_prob_distrib** ()
 $current\_estimator_j$ = **learn_base_estimator**(X_{train}, $vec\_classes_j$)
 $estimators\_list = estimators\_list \cup \{current\_estimator_j\}$

Algorithm 2: Implementation of **generate_classes_by_prob_distrib()** function
Input: $V_{train} = ((p_{11}, p_{12},\ldots,p_{1k}),\ldots, (p_{n1}, p_{n2},\ldots,p_{nk}))^T$,
n – number of instances in training set
Output: $C_{train} = (c_1,\ldots, c_n)^T$ – list of labels.

C_{train} = {}
for i= 1, ..., n:
 c_class_i = randomly choose class index in accordance with probability distribution $(p_{i1}, p_{i2},\ldots,p_{ik})$
 $C_{train} = C_{train} \cup \{c\_class_i\}$

return C_{train}

Function *learn_base_estimator*() implements learning of the base estimator (for instance, a decision tree).

The algorithm takes into account classes predicted by all m base estimators for each observed object, and final decision is made on *majority principle*. Thus, that decision actually presents the most frequent class among m predictions.

Finally, we got the following results using the algorithm, described above: **precision = 0.67**, $\boldsymbol{F_1}$**-measure = 0.701** (n_estimators = 35, size of probability part of training set = 150,000 instances).

VI. Future work

In the future we are planning to extend bagging algorithms (for instance, Random Forests) with methods that work with probability distributions over classes. Also it would be very interesting to determine a dependency between classification's quality and such parameters as a number of base estimators and the size of automatically collected objects.

In addition to that, since annotated data set was not fully labeled, probability distributions that were associated with training set are probably biased.

Finally, we are going to find out a theoretical explanation of the practical result that taking probability distributions into account improves the quality of classification.

VII. Conclusion

In this paper we proposed an approach that uses probability distributions over classes in training set.

Despite the fact that observed algorithms are relatively simple, they confirm the fact that usage of probability distributions over classes could significantly improve the quality of classification.

Acknowledgment

The paper is funded by the Russian Federal Foundation for Assistance to Small Innovative Enterprises (FASIE), project number is 0019251. The authors express their gratitude to the Foundation.

References

[1] M.J. Kearns and U.V. Vazirani, *An Introduction to Computational Learning Theory*. Cambridge, Massachusetts, London, England: MIT Press, 1994.
[2] P. Flach, *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge: University Press, 2012.
[3] T. Denoeux, L.M. Zouhal, "Handling possibilistic labels in patern classification using Evidential reasoning", *Fuzzy Sets and Systems*, vol. 122 (3), 2001, pp. 409-424.
[4] T. Denouex, "Maximum likelihood estimation from Uncertain Data in the Belief Function Framework", *IEEE Transactions on Knowledge and Data Engineering*, vol. 25 (1), 2013, pp. 119-130.
[5] T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference and Prediction.* Second Edition, Springer, 2013.
[6] O. Durandin, N. Hilal, D. Strebkov, "Automatic Arabic Dialect Identification", *in Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialogue 2016"*, 2016.
[7] N.Y. Habash, *Introduction to Arabic Natural Language Processing,* Toronto: Morgan & Claypool, 2010.
[8] L. Brieman, Random Forests, *Machine Learning*, vol. 45 (5), 2001, vol. 5-32.
[9] M.R. Segal, *Machine Learning Benchmarks and Random Forests Regression,* Tech. rep., Center for Bioinformatics & Molecular Biostatistics, UC San Francisco 2004, Web: https://escholarship.org/uc/item/35x3v9t4