Computación y Sistemas
ISSN: 1405-5546
computacion-y-sistemas@cic.ipn.mx
Instituto Politécnico Nacional
México

Alekseev, Anton; Nikolenko, Sergey
Word Embeddings for User Profiling in Online Social Networks
Computación y Sistemas, vol. 21, núm. 2, 2017, pp. 203-226
Instituto Politécnico Nacional
Distrito Federal, México

Available in: http://www.redalyc.org/articulo.oa?id=61551628004

How to cite
Complete issue
More information about this article
Journal's homepage in redalyc.org

Scientific Information System
Network of Scientific Journals from Latin America, the Caribbean, Spain and Portugal
Non-profit academic project, developed under the open access initiative

ISSN 2007-9737

Word Embeddings for User Profiling in Online Social Networks

Anton Alekseev[1,2], Sergey Nikolenko[1,2]

[1] National Research University Higher School of Economics, St. Petersburg, Russia

[2] Steklov Mathematical Institute at St. Petersburg, St. Petersburg, Russia

anton.m.alexeyev@gmail.com, sergey@logic.pdmi.ras.ru

Abstract. User profiling in social networks can be significantly augmented by using available full-text items such as posts or statuses and ratings (in the form of likes) that users give them. In this work, we apply modern natural language processing techniques based on word embeddings to several problems related to user profiling in social networks. First, we present an approach to create user profiles that measure a user's interest in various topics mined from the full texts of the items. As a result, we get a user profile that can be used, e.g., for cold start recommendations for items, targeted advertisement, and other purposes; our experiments show that the interests mining method performs on a level comparable with collaborative algorithms while at the same time being a cold start approach, i.e., it does not use the likes of an item being recommended. Second, we study the problem of predicting a user's demographic attributes such as age and gender based on his or her full-text items. We evaluate the efficiency of various age prediction algorithms based on *word2vec* word embeddings and conduct an extensive experimental evaluation, comparing these algorithms with each other and with classical baseline approaches.

Keywords. User profiling, word embeddings, distributional semantics, ranking.

1 Introduction

In the modern Web, with the advance of social interactions between users and full-scale data mining of all information related to users, user profiling has become a very important problem. In this context, user profiling means converting the recorded user behavior into a set of labels or probability distributions that capture the most important aspects of the user that can be further used for making new recommendations, providing targeted advertisement, and so on.

One could expect that user profiling can be significantly augmented with natural language processing. Much of what goes on in social networks has the form of a text, and one can use texts generated by the user him/herself such as wall posts or statuses to mine his or her interests and demographic information. The recent development of deep learning techniques for natural language processing has led to state of the art models that operate in a basically unsupervised fashion and do not require much linguistic insight; one such direction of study deals with *word embeddings*, vector representations of words that capture semantic relations between the words and can serve as an intermediate step for other models.

The contributions of this work are twofold. First, we concentrate on a novel application of word embeddings to *user profiling*, with the specific example of improving user age prediction with full-text items. Second, user profiles can incorporate knowledge such as demographic information (age, gender, location etc.) or attempt to infer it from user behavior.

However, the holy grail of user profiling is to concisely represent a user's topical interests, preferably as narrowly as possible, with obvious applications to targeted advertising and new recommendations. The methods of summarizing information about users lie at the core of many personalized search and advertisement engines

and various recommender systems. Being able to make predictions based on appropriately summarized prior user-system interaction allows, among other things, to alleviate the so-called *cold start* problem, which is one of the main problems of recommender systems: how do you recommend a new item that has not been rated before or has had very few ratings? Given user profiles and a way to match the new item to these profiles, one can make recommendations when collaborative filtering is inapplicable.

We believe that huge corpora of user-generated texts stored in forums and social networks can be used to produce interpretable, semantic user profiles and improve interests-based recommendations for full-text items. We develop new age prediction methods and algorithms for users interacting with full-text items based on distributed word representations and a novel approach to user's interests profile construction. We show improvements in demographic user profiling for these algorithms, show improved results in item recommendation over collaborative recommenders and, most interestingly, show a way to extract an interpretable user interest profile.

The paper is organized as follows. In Section 2, we survey related work on user profiling with textual information, especially inferred from social media, and word embeddings.

Section 4 is devoted to the background of our experimental study, detailing the dataset collected from the Russian social network *Odnoklassniki* and *word2vec* models we have trained for this study.

In Section 5 we show our proposed algorithms for age prediction and present comprehensive experimental results that show improved age prediction.

Section 6 defines models for semantic user profiling that are also based on word embeddings; we show sample user interest profiles and present an experimental evaluation in terms of recommending new items to the users based on their textual preferences.

We conclude with Section 7. This work is a significantly extended joint journal version of two conference papers [3, 58].

2 Related Work

2.1 User Profiling with NLP

User profiling by user behavior has had a long history in many different contexts. Previous attempts at big data user profiling without deep neural networks have leaned upon external knowledge in the form of ontologies [50] and presented a general framework for using NLP in profiling [14]. There is a large classical field of authorship analysis, attribution and author verification studies [38, 91]; we refer to surveys [16, 78, 79] for details and references.

Some works use natural language processing to perform or augment user profiling. In particular, there have been several works closer to social sciences based on available anonymized datasets that do things similar to user profiling, usually mining demographic information from texts generated by a user, and attempts to mine text to establish new information regarding a user or relations between users, a field known as *social media personal analytics*. Next, we highlight some of this research.

In [43], anonymized text messaging datasets are used to investigate the demographics of texting, while in [26], author profiling for English emails uncovers basic demographic traits (gender, age, geographic origin, level of education, and native language) and five psychometric traits based on email texts.

Several Twitter-based studies have focused on mining demographic features based on tweets [24, 32]; the work [42], for instance, does it in a weakly supervised fashion, using Facebook or Google+ profiles as distant supervision. The work [59] detects personality traits from weblog texts, while the work [5] explicitly studies lexical predictors of personality type, [10] determines demographic information by social media texts, and [67] mines user relations from online discussions; an interesting extension is [28] which attempts personality profiling of fictional characters based on the texts about them.

In [70], author profiles in social media are mined to get hidden user profile information, while in [60] metadata is used to mine author profiles;

the work [85] attempts automatic collection and summarization of personal profiles from various social networks and other sources, while [20] proposes linguistic features that help determine the natural language of a person writing in English (on a dataset of the First NLI Shared Task) and [66] determines a user's occupation by his or her tweets.

In [21, 82], the user's political preferences are determined by his or her tweets, and [40] drives it further to get the user's actual voting intentions. This kind of profiling even extends to medical issues: the work [64] attempts to screen Twitter users for depression based on their tweets. Numerous works on the topic have been published based on the results of the shared Author Profiling Tasks at digital text forensics events by PAN initiative [8, 29, 71–74, 83]; we specifically note the work [8] that uses *word2vec* clustering to get features for author profiling. Finally, there are quite a few works for determining the geographical location of a user from his or her textual activity in social networks [9, 33, 44, 68, 69, 87].

As for neural NLP models, one recent work that actually uses modern neural network-based NLP to automatically construct user profiles is [80]. There, convolutional neural networks are used to construct a joint representation of users, products, and their reviews, in particular user profiles. This results in semantic user profiles that are then used to improve sentiment classification but can probably be used for other purposes as well. A recent work [58] has used word embeddings to construct user profiles from the texts they liked in a social network; the profiles were constructed as logistic regression weights of word clusters (clustered in the semantic space of word embeddings), with a special mechanism to reduce the weights of clusters with common words and bring topical clusters to the top. In [30], a *deep semantic similarity model* (DSSM) is trained to model the "interestingness" of documents. The purpose of the model is to recommend target documents that might interest a user based on a source document which she is reading at the moment. This is mostly an information retrieval model, trained on click transitions between source and target documents; this work is similar to [81] and also uses convolutional architectures. The hierarchical neural language model from [25] with a document level and a token level can also be extended to learning user-specific vectors to represent individual preferences, which can be used to give personalized recommendations.

User profiling is a special case of user modeling. For general reviews of the field and key papers, we refer to [12, 18, 27, 36, 86]. Specific techniques that have been applied to represent user interests in content-based and hybrid recommender systems include, for example, relevance feedback and Rocchio's algorithm [47, 63], where a user profile is represented as a set of words and their weights, penalized if the retrieved textual item is uninteresting, as in [62]. Ontologies and encyclopaedic knowledge sources have been used, e.g., in Quickstep and Foxtrot systems [51] that recommend papers based on browsing history, automatically classify the topics of a paper and make use of relations between the topics in ontology to obtain their similarity; rank is computed based on the correlation of user profile topics and estimated paper topics. Nearest neighbours are often used in such systems; e.g., DailyLearner [11] stores tf-idf representations of recently liked stories in a short-term memory component, using it to recommend new stories [47, 63]. Decision rules have been used, e.g., in the RIPPER system [7, 22], where rules are a conjunction of several tests against items features. Interpretable predicted user characteristics are also often utilized in practice; cf., e.g., Yandex.Crypta.

3 Word Embeddings

Recent advances in distributed word representations have made it into a method of choice for modern natural language processing [31]. Distributed word representations are models that map each word occurring in the dictionary to a Euclidean space, attempting to capture semantic relationships between the words as geometric relationships in the Euclidean space. In a classical word embedding model, one first constructs a vocabulary with one-hot representations of individual words, where each word corresponds to its own dimension, and then trains representations

for individual words starting from there, basically as a dimensionality reduction problem. For this purpose, researchers have usually employed a model with one hidden layer that attempts to predict the next word based on a window of several preceding words. Then representations learned at the hidden layer are taken to be the word's features.

The *word2vec* embeddings come in two flavors, both introduced in [52]: *Continuous Bag-of-Words* (CBOW) and *skip-gram*. During its learning, a CBOW model is trying to reconstruct the words from their contexts, while the skip-gram model operates inversely, predicting the context from the word. The idea of word embeddings has been applied back to language modeling in [53, 54, 56], and starting from the works of Mikolov et al. [52, 55], word representations have been used for numerous NLP problems, including text classification, extraction of sentiment lexicons, part-of-speech tagging, syntactic parsing etc.

Another important model for word embeddings is *Glove* (GLObal VEctors for word representations) [65].

Efficient and/or more stable algorithms for training word embeddings have been developed in [48, 49, 52, 57].

4 Background

4.1 Datasets

For this project, we have obtained a large dataset from the *Odnoklassniki* social network. The dataset has been created as follows:

— the dataset began with 486 seed users;

— for these users, their sets of friends have been extracted;

— then the friends of these friends; as a result, the dataset contains a neighborhood of depth 2 in the social graph for the original seed users.

As a result, the dataset contains information on 868,126 users of the *Odnoklassniki* social network. In particular, it contains the following data:

— demographic information on 868,126 users of the network: gender, age, and region (region info may be imprecise since there is no such explicit field in the user's profile, the region is determined by the IP addresses from which the user has logged in most often);

— the social graph that defines the "friendship" relation and contains (and indicates) several different type of links: "friend", "love", "spouse", "parent", and so on; all users with known demographic data are also present in the social graph;

— history of logins for individual users;

— data on the "likes" ("class!" marks) a user has given to other users' statuses and posts in various groups;

— texts of user posts and group statuses that have been liked by these selected users.

The mean age of all users was 31.39 years; the age distribution is shown on Fig.1. Note that there are quite a lot of users with implausible ages (ages 2 and 3, age higher than 100 years); since the user specifies the age by himself/herself, this probably represents missing, incorrect, or purposefully distorted data. Note that this is an important point for the relevance of our research: when a user has not specified his/her age, or when a user has specified an obviously incorrect age, we still need to predict his or her age in order to give age-related recommendations and enroll the user into age cohorts. For the experiments, however, we have removed from the dataset all ages below 10 and above 80 since they are likely to correspond to faulty/missing information.

Fig.2 shows the distribution of the number of friends in the *Odnoklassniki* dataset; interestingly, while the usual Pareto distribution (straight line on a log-log plot) picks up after about 100 friends, it actually increases before that point. This is probably an artifact of the data collection: naturally, the social circle (neighborhood of depth 2) of a predefined set of seed users will contain few isolated or nearly isolated users.

We began evaluation with the entire dataset as outlined above, what is called below the "extended"

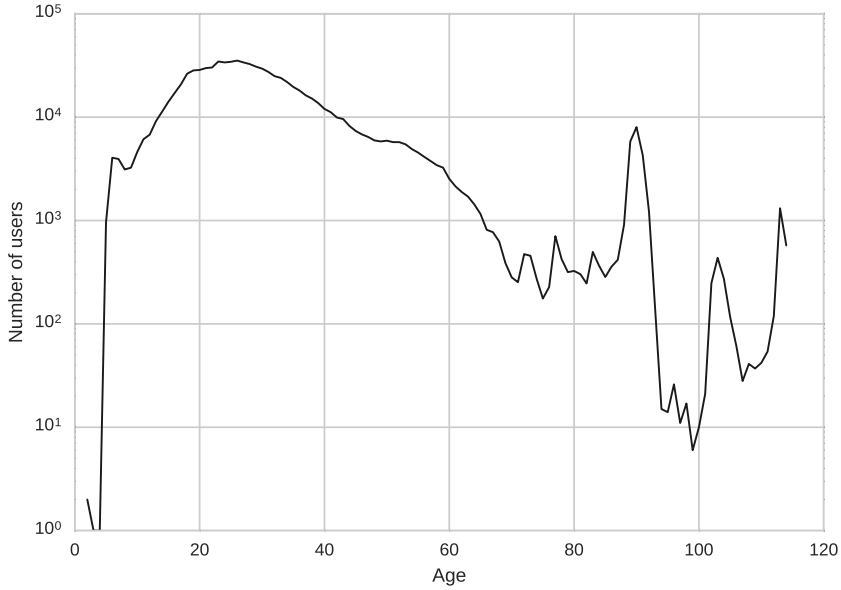


Fig. 1. Age distribution in the *Odnoklassniki* dataset

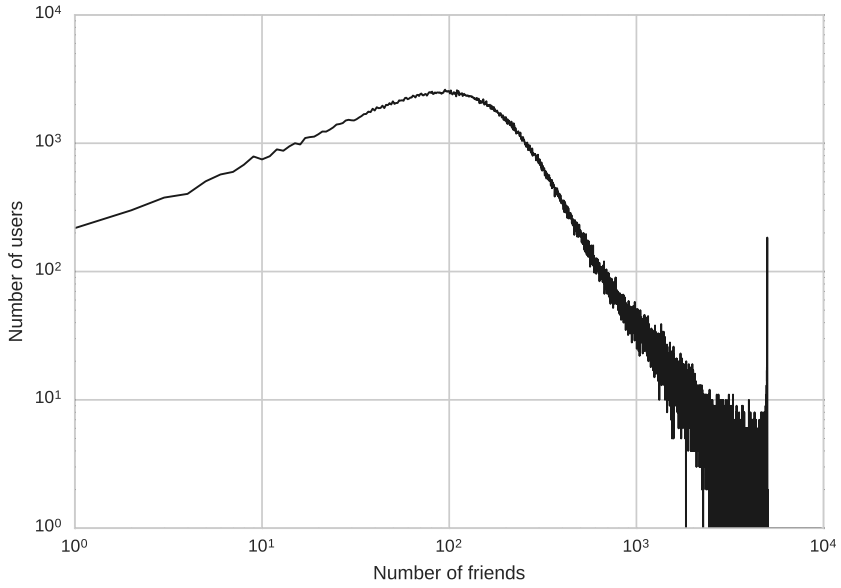


Fig. 2. Number of friends distribution in the *Odnoklassniki* dataset

dataset. However, in order to perform more experiments, be more flexible, and not get bogged down in the technicalities of fitting huge datasets into available hardware, we have also prepared a smaller "basic" dataset that we performed some experiments on.

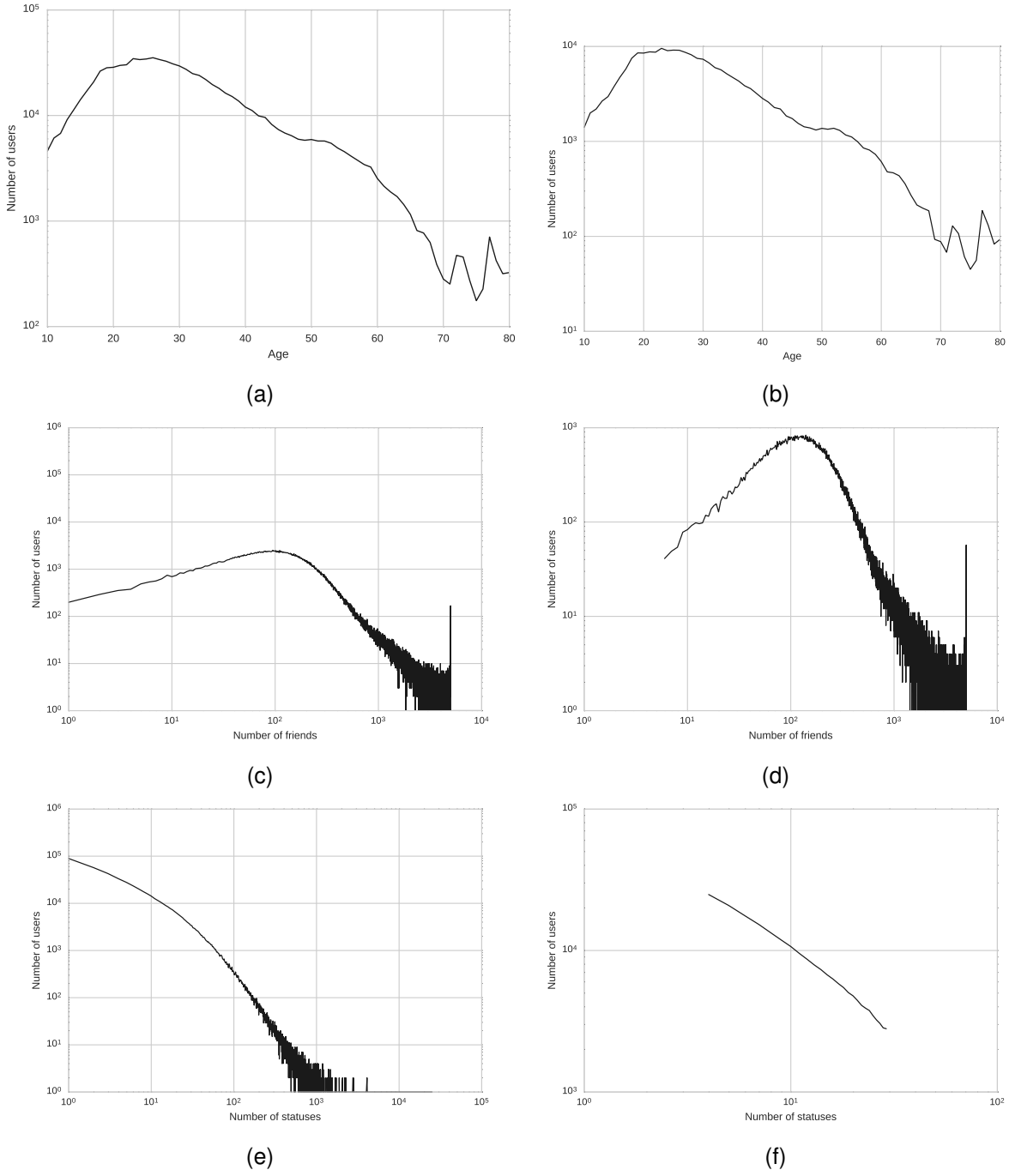


Fig. 3. Basic distributions for the datasets: (a) age distribution, extended, (b) age distribution, basic, (c) friends distribution, extended, (d) friends distribution, basic, (e) statuses distribution, extended, (f) statuses distribution, basic.

Table 1. Basic statistics for extended and basic datasets

Dataset	Training set		Test set	
	Users	Statuses	Users	Statuses
Extended	661206	10880321	165301	2704883
Basic	170856	2014983	42713	503150

Table 2. Word2vec models trained on large Russian corpora; the columns are: dimension d, window size w, number of negative samples n, vocabulary threshold v, and the resulting model size

Type	d	w	n	v	Size
CBOW	100	11	10	20	1.3G
CBOW	100	11	10	30	0.97G
skip-gram	100	11	10	30	0.97G
skip-gram	200	11	1	20	1.3G
CBOW	200	11	1	30	2.0G
skip-gram	200	11	1	30	2.0G
CBOW	300	11	1	30	2.9G
skip-gram	300	11	1	30	2.9G

The basic dataset preserves most properties of the extended dataset; the only difference is that we have filtered the users to have at least 5 and at most 300 statuses. This has let us cut off a relatively small number of highly prolific writers (or, to be more precise, prolific reposters), significantly reducing the total number of statuses, and cut off the long tail of users with very few statuses, while still preserving important properties of the data.

The basic statistics for the two datasets are shown on Table 1, and Fig. 3 indicates that all basic distributions such as age and number of friends are very similar for the two datasets, except, naturally, the distribution of the number of statuses. Both datasets were split into training and test sets randomly in the 80:20 proportion.

4.2 Word2vec Models

As a dataset for word embeddings, we have used a large Russian-language corpus (the largest we know) with about 14G tokens in 2.5M documents [4,61]. This corpus includes the Russian *Wikipedia* (1.15M documents, 238M tokens), automated Web crawl data (890K documents, 568M tokens), (main part) a huge *lib.rus.ec* library corpus (234K documents, 12.9G tokens), and, finally, user statuses and group posts from the *Odnoklassniki* social network, as described above (excluding test data used later). All of this has let us obtain what we believe to be an unprecedented quality of the resulting representations. We refer to [4, 61] for more details on the training data.

We have used continuous bag-of-words (CBOW) and skip n-gram *word2vec* models trained on a single NVidia Titan X GPU with the currently fastest *word2vec* implementation ported to CUDA (`https://github.com/ChenglongChen/word2vec_cbow`). Our previous experiments have suggested that vector sizes in the low hundreds and window size of 11 words are the best parameters on this dataset. In total, so far in the experiments we have used eight different *word2vec* models whose parameters are shown in Table 2; the models differ in the type (CBOW or skip-gram), dimension of word vectors d, window size w (later omitted since $w = 11$ in all models), number of negative samples n in the training, and vocabulary threshold v that controls the size of the vocabulary (a lower threshold means more words get vectors, but words with few occurrences will not have enough training data and might have a random-like, meaningless vector). Note also that every model can come in a "raw" form, as trained, and a normalized form where all vectors are normalized to Euclidean length 1.

5 Demographic User Profiling with Word Embeddings

5.1 User Age Prediction Algorithms

In this section, we propose several relatively simple algorithms that operate on word embeddings of the words in social network statuses of the users, aiming to predict a user's age from his or her writing.

First, folklore among social network researchers says that to predict a user's age it is usually sufficient to take the mean age of his or her friends: it will predict the age with outstanding accuracy. We have tested this hypothesis on the *Odnoklassniki* dataset. To investigate, we have trained the following models:

(1) MEANAGE: predict age with the mean of friends' ages and the global mean if no friends ages are known;

(2) LINEARREGR: linear regression with a single feature (mean friends age);

(3) ELASTICNET: elastic net regressor with a single feature (mean friends age);

(4) GRADBOOST: gradient boosting with a single feature (mean friends age).

Results of these simple models are shown in Table 3 in two variations: basic, where we substitute zeros instead of missing features (when there are no friends' ages) and "nonzero", where we train and test only on a subset of data with nonzero features (at least one friend with known age). It appears that LINEARREGR performs worse than MEANAGE in its first variation because linear regression cannot implement the condition "if the feature is zero (default value in the absence of neighbors) do something completely different", and GRADBOOST is noticeably better because it is powerful enough to handle such case-by-case conditions.

However, we should note that the errors here are quite significant: in terms of MAE, we are more than nine years off on average even if we restrict ourselves to cases with friends with known ages. Hence, we expect that subsequent work is not meaningless and can bring substantial improvements.

Note that while the idea to use the sum and/or mean of word embeddings to represent a sentence/paragraph is, indeed, the simplest idea for the representation of a larger chunk of text, due to the geometric properties of the *word2vec* and GloVe models this idea is not as naive as it sounds. This approach has been used as a baseline in [41] but was proposed as a reasonable method for short phrases in [55] and has been shown to be effective for document summarization in [37].

Thus, we propose three basic algorithms:

(1) MEANVEC: train on mean vectors of all statuses for a user;

(2) LARGESTCLUSTER: train on the centroid of the largest cluster of statuses;

(3) ALLMEANV: train on every status independently, with the mean vector of a specific status and the mean age of friends as features and the user's demography as target; at the testing stage, we compute predictions for every status and average the predictions.

The MEANVEC algorithm simply computes the mean vector of all statuses and adds it as features to the classification/regression model. Formally speaking, we introduce the following notation:

— W is the vocabulary, with words $w \in W$;

— U is the set of users, a user will usually be denoted as $u \in U$;

— S_u is the set of texts "belonging to" user u (either written by u or liked by him/her), with a single text usually denoted as $s \in S_u$; the s stands for either "string" or, more specifically, "status";

— $\boldsymbol{v}_w^m$ is the vector (word embedding) of word w in model m (we will omit the superscript when it is not important or clear from context);

— $\bar{\boldsymbol{v}}_A = \frac{1}{|A|}\sum_{w\in A}\boldsymbol{v}_w$ is the mean vector of a set of word embeddings A;

— MAF_u is the mean age of the friends of a user $u \in U$; in the algorithms, this is the only feature we use from the social graph.

In this notation, the MEANVEC algorithm operates as follows: for a machine learning (regression for age) algorithm ML,

(1) for every user $u \in U$:
   — for every status $s \in S_u$, compute its mean vector $\bar{\boldsymbol{v}}_s = \frac{1}{|s|}\sum_{w\in S}\boldsymbol{v}_w$;
   — compute the mean vector of all statuses $\bar{\boldsymbol{v}}_u = \frac{1}{|S_u|}\sum_{s\in S_u}\bar{\boldsymbol{v}}_s$;

(2) train ML with features $(\mathrm{MAF}_u, \bar{\boldsymbol{v}}_u)$ for every $u \in U$.

Table 3. Baseline results: predictions by mean age of the friends

Model	Train		Test		Train, nonzero		Test, nonzero	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
MEANAGE	9.701	6.725	9.661	6.707	8.833	5.865	8.787	5.840
LINEARREGR	11.252	8.659	11.226	8.650	8.794	5.951	8.752	5.930
ELASTICNET	11.251	8.660	11.226	8.651	8.795	5.969	8.752	5.948
GRADBOOST	9.602	6.743	9.569	6.730	8.683	5.879	8.645	5.861

The LARGESTCLUSTER algorithm operates as follows: for a machine learning algorithm ML,

(1) for every user $u \in U$:
   — for every status $s \in S_u$, compute its mean vector $\bar{\boldsymbol{v}}_s = \frac{1}{|s|} \sum_{w \in S} \boldsymbol{v}_w$;
   — cluster the set of vectors $\{\bar{\boldsymbol{v}}_s \mid s \in S\}$ into two clusters with agglomerative clustering; denote by $C \subseteq S$ the larger cluster;
   — compute the mean vector of statuses from C $\bar{\boldsymbol{c}}_u = \frac{1}{|C|} \sum_{s \in C} \bar{\boldsymbol{v}}_s$;
(2) train ML with features $(\mathrm{MAF}_u, \bar{\boldsymbol{c}}_u)$ for every $u \in U$.

Prior experiments (e.g. see 6.3) showed that clustering *word2vec* representations may yield semantically related groups of words/n-grams, and it appears natural to try a similar approach to statuses representations. Hence, the largest cluster could be expected to be the most descriptive.

The ALLMEANV algorithm operates as follows: for a machine learning algorithm ML,

(1) for every user $u \in U$ and every status $s \in S_u$, compute its mean vector $\bar{\boldsymbol{v}}_s = \frac{1}{|s|} \sum_{w \in S} \boldsymbol{v}_w$;
(2) train ML with features $(\mathrm{MAF}_u, \bar{\boldsymbol{v}}_s)$ for every $u \in U$ and $s \in S_u$;
(3) on the prediction stage, for a user $u \in U_{\mathrm{test}}$:
   — for every status $s \in S_u$, compute its mean vector $\bar{\boldsymbol{v}}_s = \frac{1}{|s|} \sum_{w \in S} \boldsymbol{v}_w$;
   — predict the age for this status, $a_s = \mathrm{ML}(\mathrm{MAF}_u, \bar{\boldsymbol{v}}_s)$;
   — return the average predicted age, $a = \frac{1}{|S_u|} \sum_{s \in S_u} \mathrm{ML}(\mathrm{MAF}_u, \bar{\boldsymbol{v}}_s)$.

5.2 Evaluating User Age Prediction

In the first experiment, we took the simplest MEANVEC algorithm and compared how various *word2vec* models perform. The results are shown in Table 4. We can draw the following conclusions:

— naturally, the MEANAGE algorithm does not care about *word2vec* at all, it is only included as a sanity check;

— *word2vec* models do help all models, both linear and GRADBOOST – compare these results with Table 3;

— it appears that CBOW models outperform skip-gram models in this task (quite significantly);

— by increasing the dimension d, we also get some improvements, but these improvements are rather small;

— a decrease in v, although it makes the *word2vec* model significantly larger and longer to train, has absolutely no effect on the end result.

Generally speaking, these conclusions mean that for the purposes of demographic analysis and similar problems we can concentrate on relatively small *word2vec* models, with dimensions 100 or 200, and perhaps further increase v, which would lead to much smaller models and faster training.

In the second experiment here, we have compared raw and normalized *word2vec* models in the same setting; some of the results are shown in Table 5; for convenience, raw and normalized versions are shown immediately next to each other. The results are rather interesting: the 'farther' the classifier is from linear models, the better normalized versions are. For LINEARREGR, raw vectors slightly outperform normalized ones,

for ElasticNet there is almost no difference, and GradBoost makes (sometimes significantly) better use of the normalized versions. This result can probably be attributed to the fact that while normalized vectors are indeed usually recommended for use, raw vectors can have larger absolute values, including rather large outliers, and simple linear models are better at picking on larger absolute values. Still, the conclusion is to use mostly normalized models in the future since we are after the best model rather than the best linear regression.

The next step was to compare baseline algorithms with each other. Table 6 shows the comparison results between MeanVec and LargestCluster algorithms (marked MV and LC) on the original (extended) dataset, shown for a selection of normalized *word2vec* models.

Interestingly, the LargestCluster algorithm invariably loses to MeanVec in all experiments. One possible reason for this might be that the largest cluster of all statuses turns out in many cases to be the least meaningful (e.g., consisting of similar reposts from an online game or of extremely brief statuses, e.g., consisting of a single smiley); we have verified this idea with a direct examination of the data but believe that in the future, variations on the idea of clustering statuses might yet prove to be useful.

This comparison has been performed on the smaller "basic" dataset that we have presented above. Results of this comparison are shown in Table 7, which marks the MeanVec, LargestCluster, and AllMeanV algorithms as MV, LC, and AV respectively.

As for the results, the LargestCluster algorithm, again, loses in almost all cases to both MeanVec and AllMeanV. What is much more interesting, however, is that AllMeanV, while performing roughly on par with MeanVec in LinearRegr and ElasticNet, begins to lose significantly to MeanVec and even LargestCluster when we use GradBoost as the classifier. This result was quite surprising since we expected that more data and more detailed status vectors (individual for each status rather than averaged over all statuses of a user) will actually bring an improvement. One possible reason for this behavior is that in passing from MeanVec to AllMeanV we have, in essence, "moved" the averaging from the semantic space of word embeddings to averaging prediction results. Hence, this result can be interpreted as showing that simple averaging works very well in the semantic space (this is not surprising given that many semantic relations become linear in the space of embeddings), even better than building an ensemble of predictions from individual statuses afterwards.

5.3 *Word2vec* Trained on Different Data

Another interesting question is whether to use generic *word2vec* models trained on large text corpora externally or train word embeddings specifically for this problem. To answer this question, we have trained *word2vec* models on the user statuses and group posts themselves with the *gensim* library. Table 8 shows a comparison for our three basic classifiers, LinearRegr, ElasticNet, and GradBoost, for the MeanVec algorithm with these "local" word embeddings and "global" word embeddings trained externally (they were used in all previous experiments). We see that while the difference for ElasticNet is nonexistent, both LinearRegr and GradBoost consistently make better use of "local" *word2vec* models. Hence, in future studies we recommend to train word embeddings locally or fine-tune global embeddings with the local dataset.

6 Mining User Interests

6.1 Problem Setting

Apart from demographic predictions, a harder and arguably even more commercially attractive task of user profiling is to concisely represent a user's topical interests, preferably as narrowly as possible. The methods of summarizing information about users lie at the core of many personalized search and advertisement engines and various recommender systems. Being able to make predictions based on appropriately summarized prior user-system interaction allows, among other things, to alleviate the so-called *cold start* problem,

Table 4. A comparison of *word2vec* models, extended dataset, MEANVEC algorithm

Word2vec params				Train		Test		Train, nonzero		Test, nonzero	
type	d	n	v	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
MEANAGE											
				9.672	6.707	9.668	6.711	8.778	5.835	8.800	5.848
LINEARREGR											
cbow	100	10	20	10.401	7.776	10.397	7.785	8.514	5.792	8.541	5.810
cbow	100	10	30	10.402	7.777	10.396	7.784	8.515	5.791	8.539	5.808
skip	100	10	30	10.818	8.219	10.813	8.232	8.624	5.863	8.645	5.879
skip	200	1	20	10.738	8.146	10.726	8.151	8.593	5.847	8.613	5.859
cbow	200	1	30	10.355	7.737	10.349	7.743	8.497	5.782	8.520	5.798
skip	200	1	30	10.735	8.143	10.724	8.148	8.592	5.846	8.613	5.859
cbow	300	1	30	10.338	7.722	10.329	7.727	8.492	5.779	8.512	5.794
skip	300	1	30	10.689	8.088	10.675	8.096	8.583	5.837	8.601	5.854
ELASTICNET											
cbow	100	10	20	10.810	8.208	10.799	8.217	8.694	5.903	8.719	5.921
cbow	100	10	30	10.806	8.203	10.795	8.212	8.702	5.909	8.726	5.926
skip	100	10	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
skip	200	1	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
cbow	200	1	30	10.949	8.349	10.937	8.359	8.736	5.934	8.760	5.951
skip	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
cbow	300	1	30	11.026	8.433	11.017	8.445	8.741	5.938	8.766	5.956
skip	300	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
GRADBOOST											
cbow	100	10	20	9.089	6.352	9.065	6.344	8.399	5.697	8.394	5.699
cbow	100	10	30	9.093	6.356	9.066	6.345	8.401	5.699	8.395	5.700
skip	100	10	30	9.294	6.527	9.277	6.529	8.495	5.766	8.491	5.770
skip	200	1	20	9.363	6.580	9.342	6.576	8.519	5.785	8.512	5.785
cbow	200	1	30	9.067	6.341	9.043	6.333	8.383	5.682	8.377	5.683
skip	200	1	30	9.365	6.583	9.344	6.580	8.520	5.784	8.512	5.785
cbow	300	1	30	9.048	6.323	9.025	6.316	8.380	5.683	8.371	5.681
skip	300	1	30	9.387	6.596	9.367	6.595	8.536	5.799	8.532	5.804

which is one of the main problems of many recommender systems: how do you recommend a new item that has not been rated before or has had very few ratings? Given user profiles and a way to match the new item to these profiles, one can make recommendations when collaborative filtering is inapplicable.

This motivation ties in well with full-text recommendations. When users interact with items that have actual text associated with them, this allows for a possibility to infer topical user profiles based on automated mining of the texts they interact with. This problem has become especially relevant in recent years due to the growth of the social Web, where users interact with various texts all the time, not only reading but actively rating them.

As for possible solutions, recent advances in natural language understanding, especially in distributional semantics, provide many promising new methods for this problem. This is precisely the path that we take in the second part of the work, using topical clusters based on distributed word representations to construct user profiles.

In this work, we propose a novel method for user profiling in full-text recommender systems, constructing a user profile as an interpretable summary of the user's interests that can also be utilized for recommending new items solely based on the prior state of the system.

6.2 Brief Outline of the Approach

First, we cluster all word representations trained on an external corpus. We have obtained high-quality clusters that are easy to interpret as possible indicators of user's interests, so they were chosen to serve as a basis for user profiling; a user

Table 5. A comparison of *word2vec* models with their normalized versions, extended dataset, MEANVEC algorithm

Word2vec params				Train		Test		Train, nonzero		Test, nonzero	
type	d	n	v	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
LINEARREGR											
cbow	100	10	20	10.402	7.777	10.396	7.784	8.515	5.791	8.539	5.808
n.cbow	100	10	20	10.426	7.807	10.418	7.807	8.535	5.807	8.560	5.824
skip	100	10	30	10.738	8.146	10.726	8.151	8.593	5.847	8.613	5.859
n.skip	100	10	30	10.753	8.156	10.736	8.159	8.601	5.853	8.620	5.864
skip	200	1	20	10.355	7.737	10.349	7.743	8.497	5.782	8.520	5.798
n.skip	200	1	20	10.363	7.748	10.351	7.746	8.512	5.795	8.532	5.808
cbow	200	1	30	10.735	8.143	10.724	8.148	8.592	5.846	8.613	5.859
n.cbow	200	1	30	10.750	8.152	10.733	8.155	8.601	5.853	8.620	5.864
skip	200	1	30	10.338	7.722	10.329	7.727	8.492	5.779	8.512	5.794
n.skip	200	1	30	10.333	7.720	10.319	7.717	8.501	5.789	8.518	5.800
cbow	300	1	30	10.689	8.088	10.675	8.096	8.583	5.837	8.601	5.854
n.cbow	300	1	30	10.687	8.083	10.668	8.084	8.586	5.840	8.603	5.853
ELASTICNET											
cbow	100	10	20	10.806	8.203	10.795	8.212	8.702	5.909	8.726	5.926
n.cbow	100	10	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
skip	100	10	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
n.skip	100	10	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
skip	200	1	20	10.949	8.349	10.937	8.359	8.736	5.934	8.760	5.951
n.skip	200	1	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
cbow	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
n.cbow	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
skip	200	1	30	11.026	8.433	11.017	8.445	8.741	5.938	8.766	5.956
n.skip	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
cbow	300	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
n.cbow	300	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
GRADBOOST											
cbow	100	10	20	9.093	6.356	9.066	6.345	8.401	5.699	8.395	5.700
n.cbow	100	10	20	9.043	6.320	9.020	6.313	8.375	5.680	8.367	5.677
skip	100	10	30	9.363	6.580	9.342	6.576	8.519	5.785	8.512	5.785
n.skip	100	10	30	9.204	6.458	9.177	6.449	8.456	5.742	8.448	5.742
skip	200	1	20	9.067	6.341	9.043	6.333	8.383	5.682	8.377	5.683
n.skip	200	1	20	8.998	6.290	8.973	6.281	8.350	5.660	8.337	5.658
cbow	200	1	30	9.365	6.583	9.344	6.580	8.520	5.784	8.512	5.785
n.cbow	200	1	30	9.204	6.457	9.176	6.447	8.455	5.743	8.448	5.743
skip	200	1	30	9.048	6.323	9.025	6.316	8.380	5.683	8.371	5.681
n.skip	200	1	30	8.983	6.274	8.965	6.271	8.341	5.656	8.333	5.656
cbow	300	1	30	9.387	6.596	9.367	6.595	8.536	5.799	8.532	5.804
n.cbow	300	1	30	9.172	6.438	9.150	6.430	8.442	5.732	8.430	5.730

would be characterized by his or her affinity to these clusters.

For the recommender system, we used a large dataset from the "Odnoklassniki" online social network; we used group posts (texts in online communities written by their members) and individual user posts (texts published by a user on his/her profile page) as full-text items and user likes for these posts as ratings.

There are two important obstacles along this way.

1. First, the dataset contains only positive signals from the users (likes), which is common in real life recommender systems but which makes it hard to train.

   While recommender systems based on such implicit information do exist, e.g., recommender systems based on max-margin nonnegative matrix factorization [39], it is unclear how to adapt them to full-text recommendation and user profiles in the semantic space.

2. Whatever technique one tries for the problem,

Table 6. A comparison of the MEANVEC and LARGESTCLUSTER algorithms on the extended dataset for various normalized *word2vec* models

	Word2vec params				Train		Test		Train, nonzero		Test, nonzero	
	type	d	n	v	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
LINEARREGR												
MV	cbow	100	10	20	10.401	7.776	10.397	7.785	8.514	5.792	8.541	5.810
LC	cbow	100	10	20	10.548	7.926	10.542	7.929	8.573	5.834	8.601	5.852
MV	cbow	100	10	30	10.402	7.777	10.396	7.784	8.515	5.791	8.539	5.808
LC	cbow	100	10	30	10.553	7.933	10.551	7.939	8.574	5.832	8.603	5.853
MV	skip	200	1	20	10.738	8.146	10.726	8.151	8.593	5.847	8.613	5.859
LC	skip	200	1	20	10.849	8.251	10.833	8.255	8.629	5.870	8.650	5.883
MV	cbow	200	1	30	10.355	7.737	10.349	7.743	8.497	5.782	8.520	5.798
LC	cbow	200	1	30	10.493	7.878	10.494	7.886	8.554	5.822	8.581	5.839
ELASTICNET												
MV	cbow	100	10	20	10.810	8.208	10.799	8.217	8.694	5.903	8.719	5.921
LC	cbow	100	10	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
MV	cbow	100	10	30	10.806	8.203	10.795	8.212	8.702	5.909	8.726	5.926
LC	cbow	100	10	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
MV	skip	200	1	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
LC	skip	200	1	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
MV	cbow	200	1	30	10.949	8.349	10.937	8.359	8.736	5.934	8.760	5.951
LC	cbow	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
GRADBOOST												
MV	cbow	100	10	20	9.089	6.352	9.065	6.344	8.399	5.697	8.394	5.699
LC	cbow	100	10	20	9.122	6.388	9.100	6.381	8.427	5.720	8.419	5.720
MV	cbow	100	10	30	9.093	6.356	9.066	6.345	8.401	5.699	8.395	5.700
LC	cbow	100	10	30	9.141	6.399	9.120	6.396	8.431	5.720	8.428	5.723
MV	skip	200	1	20	9.363	6.580	9.342	6.576	8.519	5.785	8.512	5.785
LC	skip	200	1	20	9.277	6.520	9.250	6.508	8.493	5.772	8.490	5.770
MV	cbow	200	1	30	9.067	6.341	9.043	6.333	8.383	5.682	8.377	5.683
LC	cbow	200	1	30	9.090	6.361	9.069	6.353	8.406	5.700	8.398	5.702

user profiles always tend to be dominated by clusters/topics consisting of common words that occur often in the texts of various topics, but are useless for recommendations.

The second problem was especially hard to solve; we solved it with a novel approach to user profiling based on logistic regression trained multiple times on random subsets of the dataset; this approach is described in detail below.

6.3 Clustering Word Vectors

In our experiments, we use a skip-gram *word2vec* model of dimension 500 trained on a large Russian language corpus [4, 61].

To get a finite set of possible user interests or document topics, we clustered the word vectors directly. Note that while for some other applications topic modeling [13, 84] might prove to be more useful, but in our case the basic underlying texts were too short and of too poor quality to hope for a good topic model, decisions regarding the topics would often have to be made on the basis of one or two keywords. Besides, we wanted to develop a top-down general approach that would be applicable directly even without a large and all-encompassing collection of texts available directly in the recommender system.

The embeddings of terms that occurred in our social network posts dataset resulted in 111281 vectors to be clustered in the $\mathbb{R}^{500}$ space. We have tried several methods for large-scale data clusterization, including Birch [90], DBSCAN [76], and mean shift clustering [23], but despite being generally able to process 100K+ items, these methods have proven to be not fast enough for high-dimensional data (for dimension 500 in our case), coupled with 2000 clusters.

Hence, the best option turned out to be classical k-means clustering. We applied

Table 7. A comparison of the MEANVEC, LARGESTCLUSTER, and ALLMEANV algorithms on the basic dataset, normalized *word2vec* models

	Word2vec params				Train		Test	
	type	d	n	v	RMSE	MAE	RMSE	MAE
LINEARREGR								
MV	cbow	100	10	20	8.778	6.066	8.791	6.087
LC	cbow	100	10	20	8.927	6.174	8.946	6.200
AMV	cbow	100	10	20	8.981	6.096	8.991	6.125
MV	skip	100	10	30	9.051	6.256	9.054	6.268
LC	skip	100	10	30	9.153	6.332	9.146	6.335
AMV	skip	100	10	30	9.186	6.249	9.189	6.270
MV	skip	200	1	20	8.974	6.216	8.983	6.225
LC	skip	200	1	20	9.084	6.292	9.091	6.299
AMV	skip	200	1	20	9.132	6.210	9.137	6.231
MV	cbow	200	1	30	8.734	6.035	8.736	6.052
LC	cbow	200	1	30	8.898	6.158	8.902	6.169
AMV	cbow	200	1	30	8.944	6.071	8.952	6.098
MV	skip	200	1	20	8.976	6.217	8.985	6.226
LC	skip	200	1	20	9.086	6.296	9.094	6.301
AMV	skip	200	1	20	9.133	6.211	9.139	6.232
ELASTICNET								
MV	cbow	100	10	20	9.390	6.498	9.397	6.522
LC	cbow	100	10	20	9.390	6.498	9.397	6.522
AMV	cbow	100	10	20	9.398	6.428	9.398	6.451
MV	skip	100	10	30	9.390	6.498	9.397	6.522
LC	skip	100	10	30	9.390	6.498	9.397	6.522
AMV	skip	100	10	30	9.399	6.428	9.398	6.451
MV	skip	200	1	20	9.390	6.498	9.397	6.522
LC	skip	200	1	20	9.390	6.498	9.397	6.522
AMV	skip	200	1	20	9.398	6.428	9.398	6.451
MV	cbow	200	1	30	9.390	6.498	9.397	6.522
LC	cbow	200	1	30	9.390	6.498	9.397	6.522
AMV	cbow	200	1	30	9.399	6.428	9.398	6.451
MV	skip	200	1	20	9.390	6.498	9.397	6.522
LC	skip	200	1	20	9.390	6.498	9.397	6.522
AMV	skip	200	1	20	9.399	6.428	9.398	6.451
GRADBOOST								
MV	cbow	100	10	20	8.075	5.398	8.068	5.399
LC	cbow	100	10	20	8.163	5.456	8.172	5.467
AMV	cbow	100	10	20	8.228	5.447	8.275	5.486
MV	skip	100	10	30	8.277	5.534	8.271	5.537
LC	skip	100	10	30	8.333	5.572	8.336	5.577
AMV	skip	100	10	30	8.362	5.548	8.408	5.583
MV	skip	200	1	20	8.242	5.526	8.231	5.513
LC	skip	200	1	20	8.308	5.562	8.307	5.559
AMV	skip	200	1	20	8.347	5.541	8.392	5.572
MV	cbow	200	1	30	8.032	5.368	8.026	5.366
LC	cbow	200	1	30	8.142	5.447	8.144	5.446
AMV	cbow	200	1	30	8.220	5.443	8.264	5.479
MV	skip	200	1	20	8.240	5.518	8.230	5.510
LC	skip	200	1	20	8.308	5.560	8.302	5.558
AMV	skip	200	1	20	8.347	5.542	8.392	5.573

mini-batch k-means that samples subsets of data (mini-batches) and then applies standard k-means to then: they are assigned to centroids, and centroids are "moved" to actual centers; the updates are done stochastically, after every mini-batch [77]. For initialization, we used the k-means++ approach that initializes cluster centers as far from each other as possible and then applies standard k-means to a random data subset to refine initialization [6].

Table 9 shows sample clusters together with their *idf* (inverse document frequency) values. It is clear that the most frequent clusters largely consist of common words that do not represent any specific topic that could be used for recommendations; they will be our major problem in the next section.

We begin with the following notation: D is the set of documents, C, set of clusters, T, set of all words, T_c, set of words in a cluster $c \in C$, $\mathrm{word2vec} : T \to \mathbb{R}^d$, function assigning each word its embedding, $\mathrm{df}(t)$, number of documents $t \in T$ occurs in, $\mathrm{clust} : T \to C$, function returning the cluster of a word, I_u^{like}, set of all items user u liked.

To produce user profiles, we first constructed fixed-dimensional vector representations of documents $v_{doc} \in \mathbb{R}^d$ for each document $doc \in D$, representations of clusters of documents $v_c \in \mathbb{R}^d$ for each cluster $c \in C$ based on the representations of documents, and finally representations of users $v_u \in \mathbb{R}^d$ for each user $u \in U$ based on representations of the documents they liked and the corresponding clusters; in our experiments, $d = 500$. To build vector representations, we used a straightforward approach based on averaging and idf-like weighting. Suppose that we know word2vec word embeddings for a large proportion of words in our data (not all due to typos, proper names and the like), $v_c = \frac{1}{|T_c|} \sum_{t \in T_c} \mathrm{word2vec}(t)$ for each $c \in C$. Then we define

$$\mathrm{df}_c = \sum_{t \in T_c} \mathrm{df}(t), \quad IDF_c = \log\left(\frac{\sum_{c^* \in C} \mathrm{df}_{c^*}}{\mathrm{df}_c}\right),$$

$$v_{doc} = \sum_{t \in doc} \frac{IDF_{\mathrm{clust}(t)} \cdot v_{\mathrm{clust}(t)}}{IDF^{doc}_{sum}},$$

and $IDF^{doc}_{sum} = \sum_{t\ indoc} IDF_{\mathrm{clust}(t)}$. Finally, the user representation is

$$v_u = \sum_{doc \in I^u_{liked}} \frac{\sum_{t \in doc} IDF_{\mathrm{clust}(t)} \cdot v_{\mathrm{clust}(t)}}{Z},$$

Table 8. *word2vec* trained on local and global dataset, extended dataset, MEANVEC algorithm

Word2vec params				Train		Test		Train, nonzero		Test, nonzero	
type	d	n	v	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
LINEARREGR											
Global, cbow	100	10	20	10.426	7.807	10.418	7.807	8.535	5.807	8.560	5.824
Local, cbow	100	10	100	10.321	7.703	10.302	7.701	8.518	5.783	8.505	5.794
Global, cbow	200	1	30	10.750	8.152	10.733	8.155	8.601	5.853	8.620	5.864
Local, cbow	200	10	10	10.261	7.646	10.230	7.632	8.529	5.807	8.473	5.771
ELASTICNET											
Global, cbow	100	10	20	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
Local, cbow	100	10	100	11.254	8.656	11.225	8.649	8.771	5.951	8.758	5.952
Global, cbow	200	1	30	11.239	8.641	11.229	8.653	8.741	5.938	8.766	5.956
Local, cbow	200	10	10	11.253	8.666	11.226	8.649	8.803	5.965	8.750	5.946
GRADBOOST											
Global, cbow	100	10	20	9.043	6.320	9.020	6.313	8.375	5.680	8.367	5.677
Local, cbow	100	10	100	8.986	6.270	8.951	6.263	8.348	5.651	8.318	5.649
Global, cbow	200	1	30	9.204	6.457	9.176	6.447	8.455	5.743	8.448	5.743
Local, cbow	200	10	10	8.983	6.277	8.906	6.235	8.351	5.656	8.289	5.628

Table 9. Sample *word2vec* clusters

IDF	Terms
3.276	decide family leave buy parent read case week ...
4.469	smile work answer appreciate state goal given inside brain remind ...
5.703	comment Quran culture union Kim German note interview East forum historical ...
5.902	salt pepper garlic sour-cream greens vegetables carrot cucumber ...
6.126	pain disease shock depression abortion cardiac dense muscular insomnia ...
7.608	stick axe thunder arrow sword boomerang shield spike steel armor paddle ...
8.239	lead fly move once drive run walk ...
9.650	bacteria molecule spermatozoid leukocyte chromosome mitochondria amoeba ...
11.004	scaffold gallows pardon torture quartering hanging beheading ...

where Z is a corresponding normalization value.

Then we constructed a new representation of a document, designed as a vector of cluster likelihoods $p(c \mid d)$; namely, for every document $doc \in D$ and every cluster $c \in C$ we computed

$$L(doc \mid c) = e^{\frac{-\|v_{doc} - v_c\|}{2\sigma^2}},$$

$$p(c_i|doc_j) = \frac{L(doc_j|c_i)}{\sum_{doc} L(doc|c_i)}.$$

Then, to construct the profile of a user u by his or her set of liked items I_u^{like}, we repeated the following procedure N times independently (in the experiments below, we used $N = 100$):

(i) on step k, draw a random sample from the documents the user u didn't like, taking the size of the sample equal to the number of documents the user actually liked; we denote it by $I_{u,k}^{\text{non-like}}$;

(ii) train logistic regression with the following data: I_u^{like} are the positive examples, $I_{u,k}^{\text{non-like}}$ are the negative examples, and the features are document affinities to clusters $p(c \mid d)$, $d \in I_u^{\text{like}} \cup I_{u,k}^{\text{non-like}}$;

(iii) as a result of this logistic regression, we get a set of weights $w_{u,c,k}$ for each cluster c.

Then, for every user u his or her profile is defined as the parameters of the normal distribution for every weight $\{(c, \mu_{u,c}, \sigma_{u,c}) \mid c \in C\}$, each $(\mu_{u,c}, \sigma_{u,c})$ trained on the set $\{w_{u,c,k} \mid c \in C\}$.

In other words, logistic regression here is used to approximate the probability of a like; it trains a hyperplane separating liked items from items that

have not been liked in the semantic feature space. This simple approach would be sure to fail if we simply trained liked documents against non-liked documents since the dataset is vastly imbalanced (a single user can be expected to view but a tiny fraction of all items); hence the random sampling of non-liked documents.

However, there is one more purpose to the random sampling apart from balancing the problem. We would like to solve the problem of common-word clusters, clusters that contain common words, are ubiquitous in the dataset, and tend to dominate all user profiles simply because by random chance, a user will like more than their fair share of some of common word clusters. In randomly sampling the negative examples, we get a certain distribution of "concentrations" of different clusters in the negative example. Note that:

— "topical" clusters that contain rare words will seldom occur in negative examples and thus the variance of the resulting weight distribution $\sigma_{u,c}$ with respect to these clusters will be low;

— "common word" clusters that contain words that are widely distributed across the entire dataset will sometimes appear more often and sometimes less often in the negative examples, and thus the variance of the resulting weight distribution $\sigma_{u,c}$ with respect to these clusters will be high.

Hence, this approach lets us distinguish between common word clusters and topical clusters by the value of $\sigma_{u,c}$. The higher the standard deviation, the more likely it is that the cluster consists of common words. As the final scoring metric for the user profile, we propose to use the mean weight penalized by its variance; we used $\mu - 2\sigma$ as the final score in the examples below. This scoring metric can also be thought of as the lower bound of a confidence interval for the cluster affinity. Figure 4 shows sample results of two user profiles. Note how common-words clusters have high average affinity but also high standard deviation that drags them down in the final scoring and lets topical clusters come out on top.

6.4 Recommender Algorithm and Evaluation

Here, we present an actual recommender algorithm based on the user profiles mined as shown above. This serves as both a sample application for our user profiling system and as a way to evaluate our results numerically, by comparing it to baseline recommender algorithms.

We propose the following item-based algorithm to make recommendations based on a user profile in the form $\{(c, \mu_{u,c}, \sigma_{u,c}) \mid c \in C\}$:

(1) penalize the mean of a cluster's weight distribution with its variance, $w^\star_{u,c} = \mu_{u,c} - \alpha\sigma_{u,c}$, where α is a coefficient to be tuned for a specific system;

(2) predict the probabilities of likes according to the logistic regression model with modified weights,

$$p(\text{Like} \mid doc, \boldsymbol{w}'_u) = \left(1 + \exp(\boldsymbol{v}^\top_{doc}\boldsymbol{w}^\star_u)\right)^{-1};$$

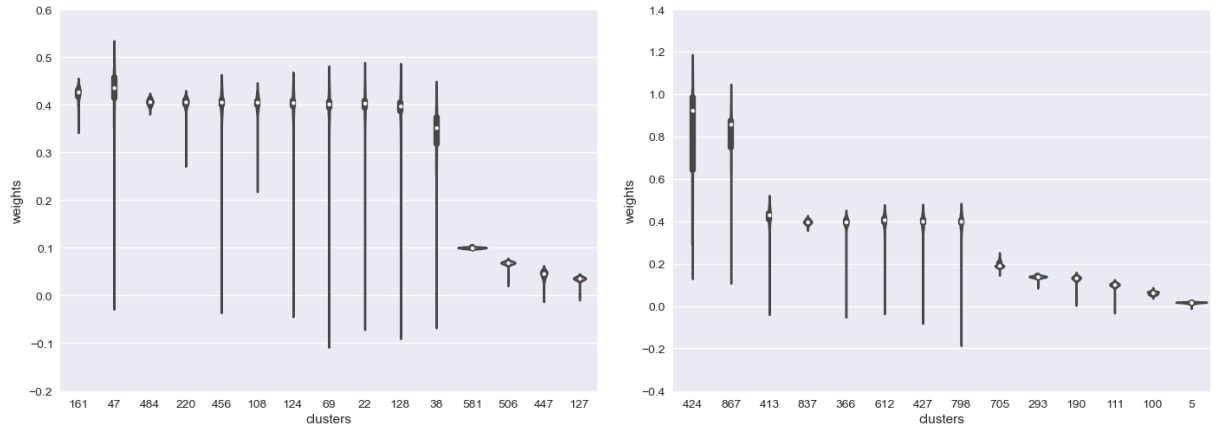
(3) rank the items according to the predicted probability.

Note that this is a cold start algorithm for the items: it does not use an item's likes at all, only the likes of a user to construct his or her profile.

We have conducted experimental evaluation with a large dataset provided by the "Odnoklassniki" social network. For the experiment, we have chosen to use posts in groups (online communities) and likes provided by the users for these posts since a post in a group, as opposed to a post in a user's profile, is likely to be evaluated by many users with different backgrounds, and the users are more likely to like it based on its topic and content rather than the person who wrote it.

Thus, the dataset consists of texts of posts in the communities (documents) and lists of users who liked the posts. The dataset contains 286K words in the vocabulary (after stemming and stop words removal), 14.3M documents (group posts), and 284.6M total tokens in these documents.

As the user set U, we chose top 2000 users with most likes from a randomly sampled subset of users (so that we get users with a lot of likes but not outliers with huge number of likes that are



#	μ	σ	Words
161	0.422	0.017	uni din tel tine adam riga van eden loc publ etc judo art professor polis...
47	0.421	0.074	Nido Josep Jordi Victor Fbio Paulinho Avito Oswaldo Oliveira Julio...
484	0.407	0.007	Virgo Aries Taurus Dey hill branch statue strand Hon patron figure Chakhon...
220	0.405	0.014	Christ soul spirit God pray purify verse forgiven salvation devote Saturday...
108	0.396	0.035	Daniel Vladimir chronicle Tver Svyatoslav Ryazan Novgorod Pskov veche...
124	0.391	0.062	ask talk call listen report try calm assuring prompt pleading convince...
69	0.386	0.081	minute managed approach this autumn song male children ages anxiety...
#	μ	σ	Words
867	0.772	0.165	hours two-hour break minute half-hour five-minute two-hour ten-hour...
424	0.833	0.202	kissing call cry silent scream laughing nod dare restrain angry slam...
837	0.399	0.010	youtube blog net mail facebook player online yandex user tor ado...
366	0.396	0.042	associate attitude seems quite horoscope ideal religious face era...
413	0.406	0.080	feel glad remember worrying offended jealous inhale pity envy suffer autumn...
427	0.385	0.073	hijack bombing raid to steal loot bomb
798	0.385	0.080	uro missile air defense mine RL submarine Vaenga Red Banner Pacific Fleet...

Fig. 4. Sample user profile produced by resampling logistic regression. Top: graphical representations of the two user profiles. Middle: sample clusters for the profile depicted on the left. Bottom: sample clusters for the profile depicted on the right

most probably bots or very uncharacteristic users). We divided their likes into disjoint training and test sets; there were 16000 likes by these users in the training set and 4797 likes in the test set.

We carried out our evaluation procedures on three algorithms: two baseline collaborative algorithms and the new algorithm described above.

User-based collaborative algorithm finds k nearest neighbors for a user and recommends documents according to the users' likes. Specifically, for each user u we build a list of k nearest neighbours $N(u)$ by cosine distance (via LSHForest) in the space of their vector representations in $\mathbb{R}^d$. Then we set the affinity between users as cosine distance between their vector representations:

$$w(u_1, u_2) = v_{u_1}^\top v_{u_2}, \quad \forall u_1, u_2 \in U.$$

Documents are ordered by the following ranking function:

$$\mathrm{rank}(u, doc) = \frac{\sum_{u' \in N(u) \cap \mathrm{Liked}(doc)} w(u, u')}{\sum_{u' \in N(u)} w(u, u')},$$

where $\mathrm{Liked}(doc)$ is the set of users who liked document doc. Thus, we rank documents

according to the weighted sum of representations of users who liked it.

Item-based collaborative algorithm finds k nearest neighbors for a document and recommends documents similar to the ones a user liked. Specifically, for each $doc \in D$ we build the set of k nearest neighbors $N(doc)$ by cosine distance in the space of vector representations for the documents, compute similarities between the documents, $w(doc_1, doc_2) = v_{doc_1}^\top v_{doc_2}$, and rank documents as

$$\mathrm{rank}(u, doc) = \frac{\sum_{doc' \in N(doc) \cap \mathrm{Like}(u)} w(doc, doc')}{\sum_{doc' \in N(doc)} w(doc, doc')},$$

where $\mathrm{Like}(u)$ is the set of documents user u liked.

In all algorithms that use k nearest neighbors approach we used an empirically chosen $k = 5$.

Finally, in our *regression-based algorithm* we recommend according to the negative-biased posterior: given a user profile $\{(\mu_{u,c}, \sigma_{u,c} \mid c \in C\}$, we rank documents according to

$$\mathrm{rank}(u, doc) = \frac{1}{1 + e^{-\sum_{c \in C} p(c|doc) w^*_{u,c}}},$$

where $w^*_{u,c} = \mu_{u,c} - \alpha\sigma_{u,c}$.

All users and documents vector representations are normalized before applying each of the algorithms above. Each of the evaluated recommender algorithms provides the ranking of documents for a given user. We build a set of all likes from test set and the same number of unliked documents for each user. It is expected that the liked documents will be ranked higher than others on average, which is a common ranking task. Hence, we used standard ranking evaluation metrics to evaluate the algorithms:

— NDCG (Normalized Discounted Cumulative Gain) is a unified metric of ranking quality [35]; the discounted cumulative gain is defined as

$$\mathrm{DCG}_p = \sum_{i=1}^{p} \frac{\mathrm{liked}_i}{\log_2(i+1)},$$

where $\mathrm{liked}_i = 1$ iff item i in the ranked list is recommended correctly, and NDCG normalizes this value to the maximal possible: $\mathrm{NDCG}_p = \mathrm{DCG}_p / \mathrm{IDCG}_p$, where IDCG_p is the perfect DCG of a ranked list with all correct items on top;

— Top1, Top5, and Top10 metrics show the share of liked documents at the first place, among the top five, and among the top ten recommendations respectively; these metrics are important for real-life recommender systems since an average user commonly views only a very small number of recommendations.

Results of our experimental evaluation are shown in Table 10. We see that the simple cold start recommender algorithm based on our user profiles performs virtually on par with collaborative algorithms that actually take into account the likes already assigned to this item. These are very good results for a cold start algorithm; note, however, that actual recommendations of full-text items in the same system are not the only or even the main purpose of our approach: the ultimate goal would be to employ user profiles to make outside recommendations for other items with textual content or tags that could be related to the interest profile, such as targeted advertising.

Another way to demonstrate that the regression method learns new things about the users and items being recommended is to show its contribution into the performance of ensembles of rankers. We used the following blending method: first, we normalized the scores obtained by the methods in the blend ($Score_m$ for each ranking method m):

$$Score^m_{norm}(doc) = \\ = \frac{Score^m(doc) - \min_{doc} Score^m(doc)}{\max_{doc} Score^m(doc) - \min_{doc} Score^m(doc)}$$

for every document doc and ranking method m, and then constructed the final scoring function as

$$Score_{blended}(doc) = e^{\sum_m e^{Score^m_{norm}(doc)} \alpha_m},$$

where α_m are blending weights to be found. We use hill climbing to tune parameters $\alpha_m \in [-1, 1]$, maximizing average NDCG for a separate validation set and finally testing performance on a production set that constituted 20% of the values. Rows 4 and 5 of Table 10 show that the blends noticeably improved upon the performance of both our regression-based approach and collaborative algorithms.

Table 10. Experimental results for the recommender algorithms

Algorithm	NDCG	Top1	Top5	Top10
User-based CF	0.7817	0.4557	1.9440	2.6557
Item-based CF	0.7904	0.4934	1.9636	2.6589
Regression-based cold start	0.7777	0.4852	1.8741	2.5960
User-based CF + regr.	0.8089	0.5508	2.0130	2.6920
Item-based CF + regr.	0.8043	0.5364	1.9834	2.6589

7 Conclusion and Future Work

In this work, we have prepared and preprocessed a huge Russian language free text dataset with a number of different sources ranging from literature to user statuses in social networks, trained a number of *word2vec* models, obtained and preprocessed a large user profiling dataset from the social network *Odnoklassniki*, suggested a number of user profiling algorithms based on *word2vec* embeddings, and performed a large-scale comparison of these algorithms and different *word2vec* models, drawing conclusions important for subsequent work on user-generated texts. We have also presented a new approach to user profiling based on logistic regression on randomly resampled subsets of items, which leads to readily interpretable user profiles; our experiments have shown that a simple cold start recommender algorithm based on user profiles produces results comparable to collaborative approaches and can be blended with them for further improvement.

While the proposed age prediction algorithms did bring certain improvements as compared to the "zero baseline" of training with the mean age of a user's friends, these improvements were not huge in absolute terms: we have been able to shave off about 0.2 years in terms of mean absolute error. Therefore, we remain optimistic that these results can be much improved in the future. In further work, we plan to

(1) develop new features for user profiling algorithms based on text embeddings (embedding larger portions of text than a word); here we hope to train a deep text understanding model for the Russian language and apply it to user profiling,

(2) develop and train a character-level word embedding model for the Russian language; we expect this model to be very important for studies of user-generated texts replete with typos, intentional misspellings, and so on.

Also, apart from developing new user profiling algorithms, we plan to investigate other variations of word embeddings. For example, one such is given by the Polyglot system [2], and a completely different direction with a graph-based model is proposed in [1].

We also note recent efforts in *word sense disambiguation* for word embeddings: the same word can have several very different meanings, and it would be natural to try to model it with several vectors in the semantic space [15, 17, 19, 34, 45, 46, 75, 88, 89]. In further work, we plan to perform an even more extensive comparison between various word embedding variations; a comparison across these models might provide valuable insight into the use of *word2vec* models for subsequent applications such as user profiling, sentiment analysis, or full-text recommendations.

Acknowledgements

We thank Dmitry Bugaichenko, Philipp Fedchin, and the *Odnoklassniki* social network (from the *Mail.Ru* holding) for the social network dataset and Alexander Panchenko and Nikolay Arefyev for general-purpose Russian-language training data.

The work of Anton Alekseev was supported by the Russian Federation grant 14.Z50.31.0030. The work of Sergey Nikolenko was supported by the Basic Research Program at the National Research University Higher School of Economics (HSE) in 2017.

References

1. **Aggarwal, C. & Zhao, P.** (**2010**). Graphical models for text: A new paradigm for text representation and processing. *Proceedings of the 33rd International ACM SIGIR, Conference on Research and Development in Information Retrieval*, SIGIR '10. ACM, New York, NY, USA, pp. 899–900. DOI: 10.1145/1835449.1835672.
2. **Al-Rfou, R., Perozzi, B., & Skiena, S.** (**2013**). Polyglot: Distributed word representations for multilingual nlp. *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, Association for Computational Linguistics, Sofia, Bulgaria, pp. 183–192.
3. **Alekseev, A. & Nikolenko, S. I.** (**2016**). Predicting the age of social network users from user-generated texts with word embeddings. *Proceeding oh 5th conference on Artificial Intelligence and Natural Language*.
4. **Arefyev, N., Panchenko, A., Lukanin, A., Lesota, O., & Romanov, P.** (**2015**). Evaluating three corpus-based semantic similarity systems for Russian. *Proceedings of International Conference on Computational Linguistics Dialogue*.
5. **Argamon, S., Dhawle, S., Koppel, M., & Pennebaker, J. W.** (**2005**). Lexical predictors of personality type, *Proceedings Joint Annual Meeting of the Interface and the Classification Society of North America*.
6. **Arthur, D. & Vassilvitskii, S.** (**2007**). K-means++: The advantages of careful seeding. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA. pp. 1027–1035.
7. **Basu, C., Hirsh, H., & Cohen, W.** (**1998**). Recommendation as classification: Using social and content-based information in recommendation. *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, AAAI '98, American Association for Artificial Intelligence, Menlo Park, CA, USA, pp. 714–720.
8. **Bayot, R., Gonçalves, T.** (**1998**). Author Profiling using SVMs and Word Embedding Averages—Notebook for PAN at CLEF 2016. *CLEF 2016 Evaluation Labs and Workshop – Working Notes Papers*, pp. 5-8, September, Évora, Portugal
9. **Berggren, M., Karlgren, J., Östling, R., & Parkvall, M.** (**2015**). Inferring the location of authors from words in their texts. *Proceedings of the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015)*, Linköping University Electronic Press, Sweden, Vilnius, Lithuania, pp. 211–218.
10. **Bergsma, S. & Van Durme, B.** (**2013**). Using conceptual class attributes to characterize social media users. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Vol. 1: Long Papers, Association for Computational Linguistics, Sofia, Bulgaria, pp. 710–720.
11. **Billsus, D. & Pazzani, M. J.** (**2000**). User modeling for adaptive news access. *User Modeling and User-Adapted Interaction*, Vol. 10, No. 2-3, pp. 147–180. DOI: 10.1023/A:1026501525781.
12. **Bjorkoy, O.** (**2010**). *User Modeling on the Web: An Exploratory Review of Recommendation Systems*, Ph.D. thesis, NTNU Trondheim.
13. **Blei, D. M., Ng, A. Y., & Jordan, M. I.** (**2003**). Latent Dirichlet allocation. *Journal of Machine Learning Research*, Vol. 3, No. 4–5, pp. 993–1022.
14. **Bloedorn, E. & Mani, I.** (**1998**). Using {NLP} for machine learning of user profiles. *Intelligent Data Analysis*, Vol. 2, No. 1–4, pp. 3–18. DOI: 10.1016/S1088-467X(98)00003-1.
15. **Boleda, G., Padó, S., & Utt, J.** (**2012**). Regular polysemy: A distributional model. *Proceedings of the First Joint Conference on Lexical and Computational Semantics, Vol. 1: Proceedings of the Main Conference and the Shared Task, and Vol. 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 151–160.
16. **Bouanani, S. E. M. E. & Kassou, I.** (**2014**). Article: Authorship analysis studies: A survey. *International Journal of Computer Applications*, Vol. 86, No. 12, pp. 22–29.
17. **Bruni, E., Tran, N. K., & Baroni, M.** (**2014**). Multimodal distributional semantics. *J. Artif. Int. Res.*, Vol. 49, No. 1, pp. 1–47.
18. **Brusilovsky, P., Kobsa, A., & Nejdl, W.**, editors (**2007**). *The Adaptive Web: Methods and Strategies of Web Personalization*. Springer-Verlag, Berlin, Heidelberg.
19. **Chen, Z., Lin, W., Chen, Q., Chen, X., Wei, S., Jiang, H., & Zhu, X.** (**2015**). Revisiting word

embedding for contrasting meaning. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Vol. 1: Long Papers, Association for Computational Linguistics, Beijing, China, pp. 106–115.

20. **Cimino, A., Dell'Orletta, F., Venturi, G., & Montemagni, S.** (**2013**). Linguistic profiling based on general–purpose features and native language identification. *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, Atlanta, Georgia, pp. 207–215.
21. **Cohen, R. & Ruths, D.** (**2013**). Classifying political orientation on twitter: It's not easy! *International AAAI Conference on Weblogs and Social Media*.
22. **Cohen, W. W.** (**1995**). Fast effective rule induction. *Twelft International Conference on Machine Learning (ML95)*, Morgan Kaufmann Publishers, pp. 115–123.
23. **Comaniciu, D. & Meer, P.** (**2002**). Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 24, No. 5, pp. 603–619. DOI: 10.1109/34.1000236.
24. **Deitrick, W., Miller, Z., Valyou, B., Dickinson, B., Munson, T., & Hu, W.** (**2012**). Gender identification on twitter using the modified balanced winnow. *Communications and Network*, Vol. 3, No. 4, pp. 189–195. DOI: 10.4236/cn.2012.43023.
25. **Djuric, N., Wu, H., Radosavljevic, V., Grbovic, M., & Bhamidipati, N.** (**2015**). Hierarchical neural language models for joint representation of streaming documents and their content. *Proceedings of the 24th International Conference on World Wide Web*, WWW '15. ACM, New York, NY, USA. DOI: 10.1145/2736277.2741643.
26. **Estival, D., Gaustad, T., Pham, S. B., Radford, W., & Hutchinson, B.** (**2007**). Author profiling for english emails. *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics (PACLING'07)*, pp. 263–272.
27. **Fischer, G.** (**2001**). User modeling in human–computer interaction. *User Modeling and User-Adapted Interaction*, Vol. 11, No.1-2, pp. 65–86. DOI: 10.1023/A:1011145532042.
28. **Flekova, L. & Gurevych, I.** (**2015**). Personality profiling of fictional characters using sense-level links between lexical resources. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Lisbon, Portugal, pp. 1805–1816.
29. **Forner, P., Navigli, R., & Tufis, D.** Clef 2013 evaluation labs and workshop–working notes papers. pp. 23-26, Valencia, Spain.
30. **Gao, J., Pantel, P., Gamon, M., He, X., Deng, L., & Shen, Y.** (**2014**). Modeling interestingness with deep neural networks. *EMNLP*.
31. **Goldberg, Y.** (**2015**). A primer on neural network models for natural language processing. *CoRR*. DOI: abs/1510.00726.
32. **Green, R. M. & Sheppard, J. W.** (**2013**). Comparing frequency-and style-based features for twitter author identification. *FLAIRS Conference*.
33. **Han, B., Cook, P., & Baldwin, T.** (**2013**). A stacking-based approach to twitter user geolocation prediction. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Association for Computational Linguistics, Sofia, Bulgaria, pp. 7–12.
34. **Huang, E. H., Socher, R., Manning, C. D., & Ng, A. Y.** (**2012**). Improving word representations via global context and multiple word prototypes. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, Vol. 1: Long Papers, Association for Computational Linguistics, pp. 873–882.
35. **Jarvelin, K. & Kekalainen, J.** (**2002**). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, Vol. 20, No. 4, pp. 422–446.
36. **Johnson, A. & Taatgen, N.** (**2005**). User modeling. *Handbook of human factors in Web design*, Lawrence Erlbaum Associates, pp. 424–439.
37. **Kågebäck, M., Mogren, O., Tahmasebi, N., & Dubhashi, D.** (**2014**). Extractive summarization using continuous vector space models. *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@ EACL*, pp. 31–39.
38. **Koppel, M., Schler, J., Argamon, S., & Messeri, E.** (**2006**). Authorship attribution with thousands of candidate. Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 659–660.

39. **Kumar, B. V., Kotsia, I., & Patras, I.** (**2012**). Max-margin non-negative matrix factorization. *Image and Vision Computing*, Vol. 30, No. 4–5, pp. 279 – 291. DOI: 10.1016/j.imavis.2012.02.010.
40. **Lampos, V., Preoţiuc-Pietro, D., & Cohn, T.** (**2013**). A user-centric model of voting intention from social media. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Vol. 1: Long Papers, Association for Computational Linguistics, Sofia, Bulgaria, pp. 993–1003.
41. **Le, Q. V. & Mikolov, T.** (**2014**). Distributed representations of sentences and documents. *CoRR*, pp. 1405.4053.
42. **Li, J., Ritter, A., & Hovy, E.** (**2014**). Weakly supervised user profile extraction from twitter. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Vol. 1: Long Papers, Association for Computational Linguistics, Baltimore, Maryland, pp. 165–174.
43. **Ling, R., Bertel, T. F., & Sundsøy, P. R.** (**2012**). The socio-demographics of texting: An analysis of traffic data. *New Media & Society*, Vol. 14, No. 2, pp. 281–298.
44. **Liu, J. & Inkpen, D.** (**2015**). Estimating user location in social media with stacked denoising auto-encoders. *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, Association for Computational Linguistics, Denver, Colorado, pp. 201–210.
45. **Liu, P., Qiu, X., & Huang, X.** (**2015**). Learning context-sensitive word embeddings with neural tensor skip-gram model. *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, AAAI Press, pp. 1284–1290.
46. **Liu, Y., Liu, Z., Chua, T.-S., & Sun, M.** (**2015**). Topical word embeddings. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, AAAI Press, pp. 2418–2424.
47. **Lops, P., Gemmis, M. D., Semeraro, G., Lops, P., Gemmis, M. D., & Semeraro, G.** (**2011**). Content-based recommender systems: State of the art and trends.
48. **Luo, Q. & Xu, W.** (**2015**). Learning word vectors efficiently using shared representations and document representations. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, AAAI Press, pp. 4180–4181.
49. **Luo, Q., Xu, W., & Guo, J.** (**2014**). A study on the cbow model's overfitting and stability. *Proceedings of the 5th International Workshop on Web-scale Knowledge Representation Retrieval & Reasoning*, Web-KR '14, ACM, New York, NY, USA, pp. 9–12. DOI: 10.1145/2663792.2663793.
50. **Middleton, S. E., Shadbolt, N. R., & De Roure, D. C.** (**2004**). Ontological user profiling in recommender systems. *ACM Transactions on Information Systems (TOIS)*, Vol. 22, No. 1, pp. 54–88.
51. **Middleton, S. E., Shadbolt, N. R., & De Roure, D. C.** (**2004**). Ontological user profiling in recommender systems. *ACM Trans. Inf. Syst.*, Vol. 22, No. 1, pp 54–88. DOI: 10.1145/963770.963773.
52. **Mikolov, T., Chen, K., Corrado, G., & Dean, J.** (**2013**). Efficient estimation of word representations in vector space. *CoRR*, pp. 1301–3781.
53. **Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., & Khudanpur, S.** (**2010**). Recurrent neural network based language model. *INTERSPEECH*, Vol. 2, pp. 3.
54. **Mikolov, T., Kombrink, S., Burget, L., vCernockỳ, J. H., & Khudanpur, S.** (**2011**). Extensions of recurrent neural network language model. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5528–5531.
55. **Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J.** (**2013**). Distributed representations of words and phrases and their compositionality. *CoRR*, pp. 1310.4546.
56. **Mnih, A. & Hinton, G. E.** (**2009**). A scalable hierarchical distributed language model. *Advances in neural information processing systems*, pp. 1081–1088.
57. **Mnih, A. & Kavukcuoglu, K.** (**2013**). Learning word embeddings efficiently with noise-contrastive estimation. **Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., & Weinberger, K. Q.**, editors, *Advances in Neural Information Processing Systems 26*, Curran Associates, Inc., pp. 2265–2273.
58. **Nikolenko, S. I. & Alekseyev, A.** (**2016**). User profiling in text-based recommender systems based on distributed word representations. *Proceedings 5th International Conference on Analysis of Images, Social Networks, and Texts*.
59. **Oberlander, J. & Nowson, S.** (**2006**). Whose thumb is it anyway?: classifying author personality from weblog text. *Proceedings of the COLING/ACL on Main conference poster sessions*, Association for Computational Linguistics, pp. 627–634.

60. **Paik, W., Yilmazel, S., Brown, E., Poulin, M., Dubon, S., & Amice, C.** (**2001**). Applying natural language processing (nlp) based metadata extraction to automatically acquire user preferences. *Proceedings of the 1st International Conference on Knowledge Capture*, K-CAP '01, ACM, New York, NY, USA. pp. 116–122. DOI: 10.1145/500737.500757.

61. **Panchenko, A., Loukachevitch, N., Ustalov, D., Paperno, D., Meyer, C. M., & Konstantinova, N.** (**2015**). Russe: The first workshop on russian semantic similarity. *Proceedings of the International Conference on Computational Linguistics and Intellectual Technologies (Dialogue)*, pp. 89–105.

62. **Pazzani, M. & Billsus, D.** (**1997**). Learning and revising user profiles: The identification of interesting web sites. *Mach. Learn.*, Vol. 27, No. 3, pp. 313–331. DOI: 10.1023/A:1007369909943.

63. **Pazzani, M. J. & Billsus, D.** (**2007**). The adaptive web. Chapter in Content-based Recommendation Systems, Springer-Verlag, Berlin, Heidelberg, pp. 325–341.

64. **Pedersen, T.** (**2015**). Screening twitter users for depression and ptsd with lexical decision lists. *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, Association for Computational Linguistics, Denver, Colorado, pp. 46–53.

65. **Pennington, J., Socher, R., & Manning, C.** (**2014**). Glove: Global vectors for word representation. *Proceedings Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, pp. 532–1543.

66. **Preoţiuc-Pietro, D., Lampos, V., & Aletras, N.** (**2015**). An analysis of the user occupational class through twitter content. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Vol. 1: Long Papers, Association for Computational Linguistics, Beijing, China, pp. 1754–1764.

67. **Qiu, M., Yang, L., & Jiang, J.** (**2013**). Mining user relations from online discussions using sentiment analysis and probabilistic matrix factorization. *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Atlanta, Georgia, pp. 401–410.

68. **Rahimi, A., Cohn, T., & Baldwin, T.** (**2015**). Twitter user geolocation using a unified text and network prediction model. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Vol. 2: Short Papers, Association for Computational Linguistics, Beijing, China, pp. 630–636.

69. **Rahimi, A., Vu, D., Cohn, T., & Baldwin, T.** (**2015**). Exploiting text and network context for geolocation of social media users. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Denver, Colorado, pp. 1362–1367.

70. **Rangel, F. & Rosso, P.** (**2016**). On the impact of emotions on author profiling. *Information Processing & Management*, Vol. 52, No. 1, pp. 73 – 92. DOI: 10.1016/j.ipm.2015.06.003.

71. **Rangel, F., Rosso, P., Moshe Koppel, M., Stamatatos, E., & Inches, G.** (**2013**). Overview of the author profiling task at PAN. *CLEF Conference on Multilingual and Multimodal Information Access Evaluation*, CELCT, pp. 352–365.

72. **Rangel, F., Rosso, P., Potthast, M., Stein, B., & Daelemans, W.** (**2015**). Overview of the 3rd author profiling task at PAN. *CLEF*.

73. **Rangel, F., Rosso, P., Potthast, M., Trenkmann, M., Stein, B., Verhoeven, B., Daeleman, W., et al.** (**2014**). Overview of the 2nd author profiling task at PAN 2014. *CEUR Workshop Proceedings*, Vol. 1180, CEUR Workshop Proceedings, pp. 898–927.

74. **Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M., & Stein, B.** (**2016**). Overview of the 4th author profiling task at PAN cross-genre evaluations. *Working Notes Papers of the CLEF*.

75. **Reisinger, J. & Mooney, R. J.** (**2010**). Multi-prototype vector-space models of word meaning. *Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 109–117.

76. **Sander, J., Ester, M., Kriegel, H.-P., & Xu, X.** (**1998**). Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data Min. Knowl. Discov.*, Vol. 2, No. 2, pp. 169–194. DOI: 10.1023/A:1009745219419.

77. **Sculley, D.** (**2010**). Web-scale k-means clustering. *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, ACM, New York, NY, USA, pp. 1177–1178. DOI: 10.1145/1772690.1772862.

78. **Stamatatos, E.** (**2009**). A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, Vol. 60, No. 3, pp. 538–556.

79. **Stamatatos, E., Daelemans, W., Verhoeven, B., Juola, P., López-López, A., Potthast, M., & Stein, B.** (**2015**). Overview of the author identification task at PAN.

80. **Tang, D., Qin, B., & Liu, T.** (**2015**). Learning semantic representations of users and products for document level sentiment classification. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Vol. 1: Long Papers, Association for Computational Linguistics, Beijing, China, pp. 1014–1023.

81. **tau Yih, W., He, X., & Meek, C.** (**2014**). Semantic parsing for single-relation question answering. *Proceedings of ACL*, Association for Computational Linguistics.

82. **Volkova, S., Coppersmith, G., & Van Durme, B.** (**2014**). Inferring user political preferences from streaming communications. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Vol. 1: Long Papers, Association for Computational Linguistics, Baltimore, Maryland, pp. 186–196.

83. **Busger op Vollenbroek, M., Carlotto, T., Kreutz, T., Medvedeva, M., Pool, C., Bjerva, J., Haagsma, H. & Nissim, M.** (**2016**). GronUP: Groningen User Profiling—Notebook for PAN at CLEF. *CLEF Evaluation Labs and Workshop – Working Notes Papers*, pp. 5-8, Évora, Portugal.

84. **Vorontsov, K., Frei, O., Apishev, M., Romov, P., Suvorova, M., & Yanina, A.** (**2015**). Non-bayesian additive regularization for multimodal topic modeling of large collections. *Proceedings Workshop on Topic Models: Post-Processing and Applications*, TM '15. ACM, New York, NY, USA, pp. 29–37. DOI: 10.1145/2809936.2809943.

85. **Wang, Z., Ll, S., Kong, F., & Zhou, G.** (**2013**). Collective personal profile summarization with social networks. *Proceedings Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Seattle, Washington, USA, pp. 715–725.

86. **Webb, G. I., Pazzani, M. J., & Billsus, D.** (**2001**). Machine learning for user modeling. *User Modeling and User-Adapted Interaction*, Vol. 11, No. 1-2, pp. 19–29. DOI: 10.1023/A:1011117102175.

87. **Wing, B. & Baldridge, J.** (**2014**). Hierarchical discriminative classification for text-based geolocation. *Proceedings Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, pp. 336–348.

88. **Wu, Z. & Giles, C. L.** (**2015**). Sense-aware semantic analysis: A multi-prototype word representation model using wikipedia. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, AAAI Press, pp. 2188–2194.

89. **Yih, W.-t., Zweig, G., & Platt, J. C.** (**2012**). Polarity inducing latent semantic analysis. *Proceedings Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 1212–1222.

90. **Zhang, T., Ramakrishnan, R., & Livny, M.** (**1996**). Birch: An efficient data clustering method for very large databases. *SIGMOD Rec.*, Vol. 25, No. 2, pp. 103–114. DOI: 10.1145/235968.233324.

91. **Zheng, R., Qin, Y., Huang, Z., & Chen, H.** (**2003**). Authorship analysis in cybercrime investigation. *Intelligence and Security Informatics*, Springer, pp. 59–73.

Article received on 14/11/2016; accepted on 17/03/2017.
Corresponding author is Anton Alekseev.