Interactive Dictionary Expansion using Neural Language Models

Alfredo Alba, Daniel Gruhl, Petar Ristoski, and Steve Welch

IBM Research Almaden, CA, US
aalba@us.ibm.com, dgruhl@us.ibm.com, petar.ristoski@ibm.com, welchs@us.ibm.com

Abstract. Dictionaries and ontologies are foundational elements of systems extracting knowledge from unstructured text. However, as new content arrives keeping dictionaries up-to-date is a crucial operation. In this paper, we propose a human-in-the-loop (HumL) dictionary expansion approach that employs a lightweight neural language model coupled with tight HumL supervision to assist the user in building and maintaining a domain-specific dictionary from an input text corpus. The approach is based on the *explore/exploit* paradigm to effectively discover new instances (explore) from the text corpus as well as predict new *"unseen"* terms not currently in the corpus using the accepted dictionary entries (exploit). We evaluate our approach on a real-world scenario in the healthcare domain, in which we construct a dictionary of adverse drug reactions from user blogs as input text corpus. The evaluation shows that using our approach the user can easily extend the input dictionary, where tight human-in-the-loop integration results in a 216% improvement in effectiveness.

1 Introduction

Dictionary expansion [12] is one area where close integration of humans into the discovery loop has been shown to enhance task performance substantially over more traditional post-adjudication. This is not surprising, as dictionary membership is often a fairly subjective judgment (e.g., should a fruit dictionary include tomatoes?) [13]. Thus even with a system which finds "similar" terms (e.g., `word2vec`) guidance is important to keep the system focused on the subject matter expert's notion of lexicon.

In this work we propose a feature agnostic approach for dictionary expansion based on lightweight neural language models, such as `word2vec` [9]. To prevent semantic drift during the dictionary expansion, we effectively include human-in-the-loop (HumL). Given an input text corpus and a set of seed examples, the proposed approach runs in two phases, *explore* and *exploit*, to identify new potential dictionary entries. The *explore* phase tries to identify similar instances to the dictionary entries that are present in the input text corpus, using term vectors from the neural language model to calculate a similarity score. The *exploit* phase tries to construct more complex multi-term phrases based on the instances

already in the input dictionary. Multi-term phrases are a challenge for word2vec style systems as they need to be "known" prior to model creation. To identify multi-term phrases, most commonly a simple phrase detection model is used, which is based on a term's co-occurrence score, i.e., terms that often appear together probably are part of the same phrase [9]. The phrase detection must be done before the model is built, and they remain unchanged after the model is built. However, depending on the domain and the task, the instances of interest evolve, or the example corpus may not be complete. For example, valid phrase combinations may simply not occur (e.g., acute joint pain may appear in the sample corpus, but for some reason chronic hip pain may not). However, these phrases are likely to occur in future texts from the same source, and thus are important to include in any entity extraction lexicon.

In the *exploit* phase, the approach generates new phrases by analyzing the single terms of the instances in the input dictionary. We use two phrase generation algorithms: (i) modify the phrases by replacing single terms with similar terms from the text corpus, e.g., "abnormal behavior" can be modified to "strange behavior"; (ii) extend the instances with terms from the text corpus that are related to the terms in the instance, e.g., *abnormal blood clotting problems* is a an adverse drug reaction, which doesn't appear as such in a large text corpus, however the instances "abnormal blood count", "blood clotting" and "clotting problems" appear several times in the corpus, which can be used to build the more complex instance. The approach allows us to construct new multi-term instances that don't appear as such in the text corpus, but there is enough statistical evidence in the corpus that such instances might be of interest for the user.

Combining the *explore* and *exploit* approaches in an unsupervised fashion (or an infrequently supervised fashion) is not particularly effective. It tends to generate many spurious results that the human subject matter expert needs to wade through. Close supervision, however, results in a much more performant system. The evaluation shows that high promptness of the HumL (tighter computer/human partnership) results in nearly perfect performance of the system, i.e., nearly all the candidates identified by the system are valid entries in the dictionary. More precisely, the experiments show that the system is 216% more effective when receiving HumL feedback after 10 identified candidates, compared to receiving HumL feedback after 500 identified candidates, while both cases require equal amount of human effort.

The rest of this paper is structured as follows. In Section 2, we give an overview of related work. In Section 3, we present our interactive dictionary expansion approach, followed by an evaluation in Section 4; We conclude with a summary and an outlook on future work.

2 Related Work

Dictionaries and ontologies are the backbone of many NLP and information retrieval systems. Hence, a lot of work in the literature focuses on identifying

new approaches for more efficient and more effective dictionary extraction from unstructured text.

Riloff and Jones [12], is one of the first works to propose an automatic iterattive approach for dictionary extraction from unstructured text. The approach uses mutual bootstrapping technique that learns extraction patterns from the seed terms and then exploits the learned extraction patterns to identify more terms that belong to the semantic category. In the following years, many similar approaches have been developed [13,3,7,2]. However, all these approaches require NLP parsing for feature extraction, and have a reliance on syntactic information for identifying quality patterns. Hence, such approaches underperform on not-so-well structured texts, like user-generated text. Furthermore, without human-in-the-loop, iterative methods can easily generate semantic drift.

One of the major challenges with concept extraction involves dealing with not-so-well structured text, given the importance of user generated content, which can prove to be extremely valuable source of information for many domains, pharmacovigilance being one of those.[1] To this end, Lee et al. [8] propose a semi-supervised model which uses a random Twitter stream as unlabeled training data and prove it successful for the recognition of Adverse Drug Reaction. Another hurdle is the fact that the dictionary to be created can be highly dependent on the task at hand, especially when dealing with positive/negative words which are highly domain-dependent [6,11]. While completely automatic techniques are highly appealing they need to be fine-tuned for every new task. We propose a human-in-the-loop approach where the "tuning" is an integral part of the process, i.e. the human works in partnership with the statistical method to drive the semantic of the task effectively and efficaciously.

Many works rely on machine learning techniques and tailor the algorithms to certain specific domains (e.g. drugs): these methods are in general expensive, requiring an annotated corpus and/or domain specific feature extraction (a comprehensive overview can be found in [10]).

Our work is closely related to *glimpse* [5] and *glimpseLD* [1]. Glimpse is a statistical algorithm for dictionary extraction based on SPOT [5] with a faster underlying matching engine. The input is a large text corpus and a set of seed examples. Starting from these it evaluates the *contexts* (the set of words surrounding an item) in which the seeds occur and identifies "good" contexts. Contexts are scored retrospectively in terms of how many "good" results they generate. All contexts are kept which have a score over a given threshold and the candidates that appear in the most "good" contexts are provided first to the HumL. The approach has been extended to *glimpseLD* [1], which is language agnostic and uses Linked Data to as a bootstrapping source. While both approaches have been proven to achieve high effectiveness for dictionary extension, both of the approaches can only identify new dictionary entries that are only present in the input text corpus. In this work, we adopt the *glimpse* computer/human part-

[1] PSB2016 is a recent benchmarking initiative on the problem `http://diego.asu.edu/psb2016/sharedtaskeval.html`.

nership architecture and extend it with the *explore/exploit* algorithm for more effective dictionary expansion.

3 Approach

The input of the algorithm is a text corpus TC and a set of dictionary seed example terms S. In the preprocessing step, we build a `word2vec` model [9], with the skip-gram implementation using TC as an input. Word2vec is a particularly computationally-efficient two-layer neural net model for learning term embeddings from raw text. The output of the model is an embedding matrix W, where each term (word or phrase) from the corpus vocabulary V_{TC} is represented as an n-dimensional vector. Projecting such latent representations of words into a lower dimensional feature space shows that semantically similar words appear closer to each other.

Our approach is based on the *explore/exploit* paradigm to effectively discover new instances (explore) from the text corpus and generate new *"unseen"* instances based on user feedback (exploit). The approach runs in iterations, where each iteration runs first the *explore* phase then the *exploit* phase. The *explore* phase uses the instances available in the input dictionary to identify similar candidates that are already present in the corpus vocabulary V_{TC}, which are then accepted or rejected by the HumL. The accepted candidates are then added to the input dictionary and are used in the *exploit* phase as well as the next *explore* iteration. During the *exploit* phase, we use the instances in the input dictionary to construct more complex phrases that might be of interest for the user.

3.1 Explore

As previously mentioned, in the word2vec feature embedding space, semantically similar words appear close to each other in the feature space. Therefore, the problem of calculating the similarity between two instances is a matter of calculating the distance between two instances in the given feature space. To do so we use the standard cosine similarity measure which is applied on the vectors of the instances. Formally, the similarity between two terms w_1 and w_2, with vectors V_1 and V_2, is calculated as the cosine similarity between the vectors V_1 and V_2:

$$sim(w_1, w_2) = \frac{V_1 \cdot V_2}{||V_1|| \cdot ||V_2||} \tag{1}$$

We calculate the similarity between the instances in the input dictionary and all the words in the corpus vocabulary V_{TC}. We sort the vocabulary in descending order using the cumulative similarity score, and choose the top-N candidates to present to the HumL. The accepted candidates are added in the input dictionary, which are then used in the *exploit* phase and the next iteration.

3.2 Exploit

In the *exploit* phase we try to identify more complex phrases that don't exist in the corpus vocabulary by analyzing the structure of the instances in the input dictionary.

This is critical to help "future proof" a lexicon against new text. For a surveillance application (e.g., drug side effects mentioned on twitter) it reduces how frequently a human needs to "tune up" the lexicon to make sure it is catching all relevant entity instances.

We use two phrase generation algorithms.

In the first approach, we first break each instance in to a set of single terms $T = \{t_1, t_2, ..., t_n\}$, then for each term t_i in T we identify a set of similar terms $TS_{t_i} = \{ts_1, ts_2, ..., ts_s\}$ in the vocabulary V_{TC} using Equation 1. In the next step, we build new phrases by replacing t_i with a term ts_i from TS_{t_i}. The new phrases are sorted based on the similarity score and the top-N are selected as candidates. For example, given the entry "abnormal behavior" the approach will identify "strange behavior", "abnormal attitude" and "strange attitude".

In the second approach, we generate new phrases by extending the instances with terms from the text corpus that are related to the terms in the instance. Related terms are terms that often share the same context, which means they often are surrounded by similar words. Given a `word2vec` model, we calculate the relatedness between two terms w_1 and w_2, as the probability $p(w_1|w_2)$ calculated using the softmax function,

$$p(w_1|w_2) = \frac{exp(v'^T_{w_1} v_{w_2})}{\sum_{w=1}^{V} exp(v'^T_w v_{w_2})}, \tag{2}$$

where v_w and v'_w are the input and the output vector of the word w, and V is the complete vocabulary of words.

As before, we first break each instance in to a set of single terms $T = \{t_1, t_2, ..., t_n\}$, then for each term t_i in T we identify a set of similar terms $TR_{t_i} = \{tr_1, tr_2, ..., tr_r\}$ in the vocabulary V_{TC} using Equation 2. In the next step, we build new phrases by appending a term tr_i from TR_{t_i} to each term t_i from T. The new phrases are sorted based on the relatedness score and the top-N are selected as candidates. For example, given the instance "clotting problems" in the input dictionary the approach first tries to identify related terms in the text corpus for "clotting". For which the top word is "blood", because in many sentences "blood clotting" appears as a phrase, which can be used to generate new instances "blood clotting problems". In the next iteration the phrase can be further extended, by identifying new related words. For example, in the top-N related words for "blood" we will find "abnormal", which can be used to generate the instance "abnormal blood clotting problems".

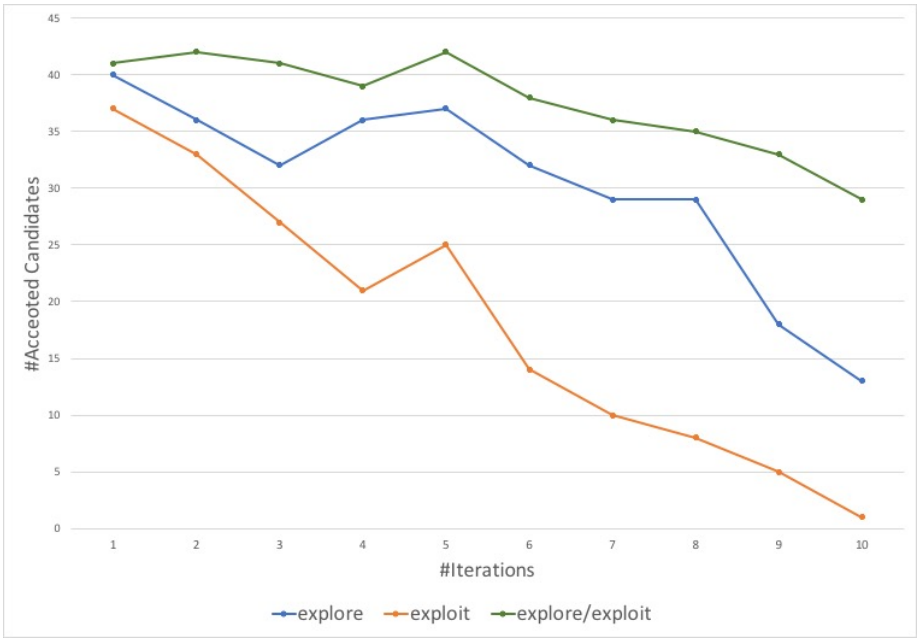

Fig. 1: Number of discovered new dictionary instances per iteration, using *explore*, *exploit*, and *explore/exploit* approaches

4 Evaluation

To evaluate our approach we conduct two experiments, i.e., (i) count the number of newly discovered dictionary entries per iteration; (ii) the impact of the promptness of the HumL on the system performance.

For the experiments we use data from the healtcare domain, specifically tackling the problem of identifying Adverse Drug Reactions in user generated data. As an input text corpus we use user blogs extracted from `http://www.askapatient.com` (a forum where patients report their experience with medication drugs). As an input set of seed examples we use a set of *203* instances referring to adverse drug events, which were labeled by a medical doctor [4].

4.1 Dictionary Growth

In this experiment we compare the performance of the *explore*, *exploit* and the *explore/exploit* approaches for discovering new dictionary instances. We run the evaluation in 10 iterations, where after each iteration we count how many new instances are discovered in the top 50 proposed candidates by the algorithm. The accepted instances are then added in the dictionary and used for the next iteration. For the *explore/exploit* approach we run *explore* to identify 25 candidates, and *exploit* to identify another 25 candidates. The results are shown in Fig. 1.

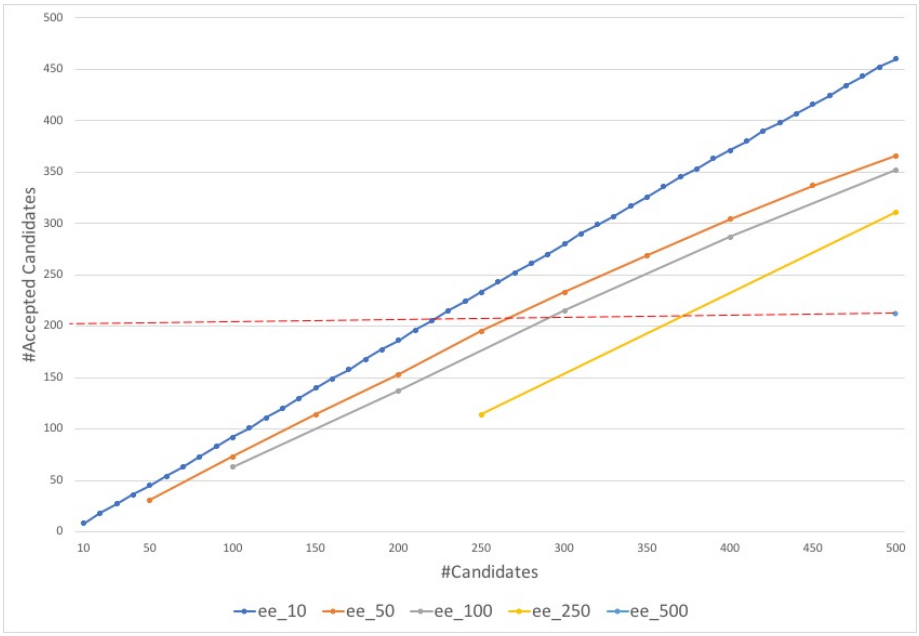

Fig. 2: Impact of the HumL promptness on the number of newly discovered dictionary instances

The results show that using the *explore/exploit* approach we are able to discover significantly more instances in each iteration compared to the other approaches. We can observe that when using the *explore* approach the number of newly discovered instances quickly decreases as the number of available instances in the whole corpus is decreasing in each iteration. When using the *exploit* approach the number of newly discovered instances sharply decreases as no new base terms are introduced, thus the *exploit* cannot generate new instances that can be added in the dictionary.

The results show that using *explore* and *exploit* alternately leads to the best performances.

4.2 Impact of the HumL on the Dictionary Growth

In this experiment we show the importance of the promptness of the HumL on the number of newly discovered instances, i.e., we evaluate if the user gives their feedback to the system sooner it will improve the performance of the system. To do so, we run the *explore/exploit* approach with different feedback intervals. The feedback interval indicates how many candidates the system needs to identify before the user gives their feedback to the system. For example, when using feedback interval of 10, the user gives their feedback after 10 candidates are identified by the system. We evaluate feedback intervals of 10, 50, 100, 250 and 500. After each iteration we count the number of accepted candidates, and

include them in the dictionary to be used for the next iteration. The results are shown in Fig 2.

The results show that the tighter the HumL integration is, the more quickly new instances are discovered. We see that with a large 500 examples feedback interval the HumL system discovers 212 new instances, but requires the human to consider 500 candidates.

A more tightly integrated system with a 10 examples feedback interval finds 212 new instances in just 23 iterations, requiring the human to consider only 230 candidates. After 50 iterations the system discovered 460 new dictionary entries, compared to only 212 new entries when using 500 examples feedback interval. That yields 216% improvement in effectivness of the system.

5 Conclusions and future work

This paper proposes an interactive dictionary expansion tool using a lightweight neural language model. Our algorithm is iterative and purely statistical, hence does not require any feature extraction beyond tokenization. It incorporates human feedback to improve performance and control semantic drift at every iteration cycle. The experiments showed high importance of tight HumL integration on discovery efficiency.

In this work, we have considered only lightweight language models, which can be efficiently built and updated on large text corpora. In future work, we will analyze more complex language neural network models, such as Recurrent Neural Networks (RNN), Long Short Term Memory Networks (LSTM), and bidirectional LSTM, which might improve the search for similar and related terms, at the expense of higher training time. Furthermore, future work will include an evaluation of the approach on multiple datasets covering different domains.

References

1. Alba, A., Coden, A., Gentile, A.L., Gruhl, D., Ristoski, P., Welch, S.: Multi-lingual concept extraction with linked data and human-in-the-loop. In: Proceedings of the Knowledge Capture Conference. p. 24. ACM (2017)
2. Ando, R.K.: Semantic lexicon construction: Learning from unlabeled data via spectral analysis. Tech. rep., IBM THOMAS J WATSON RESEARCH CENTER YORKTOWN HEIGHTS NY (2004)
3. Blohm, S., Cimiano, P.: Using the web to reduce data sparseness in pattern-based information extraction. In: PKDD 2007. pp. 18–29. Springer (2007), `https://doi.org/10.1007/978-3-540-74976-9_6`
4. Clarkson, K., Gentile, A.L., Gruhl, D., Ristoski, P., Terdiman, J., Welch, S.: User-centric ontology population
5. Coden, A., Gruhl, D., Lewis, N., Tanenblatt, M., Terdiman, J.: SPOT the drug! An unsupervised pattern matching method to extract drug names from very large clinical corpora. Proceedings - 2012 IEEE 2nd Conference on Healthcare Informatics, Imaging and Systems Biology, HISB 2012 pp. 33–39 (2012)

6. Hamilton, W.L., Clark, K., Leskovec, J., Jurafsky, D.: Inducing domain-specific sentiment lexicons from unlabeled corpora. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 595–605. Association for Computational Linguistics, Austin, Texas (November 2016), https://aclweb.org/anthology/D16-1057
7. Igo, S.P., Riloff, E.: Corpus-based semantic lexicon induction with web-based corroboration. In: Proceedings of the Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics. pp. 18–26. Association for Computational Linguistics (2009)
8. Lee, K., Qadir, A., Hasan, S.A., Datla, V., Prakash, A., Liu, J., Farri, O.: Adverse Drug Event Detection in Tweets with Semi-Supervised Convolutional Neural Networks (2017)
9. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
10. Pazienza, M.T., Pennacchiotti, M., Zanzotto, F.M.: Terminology Extraction: an analysis of linguistic and statistical approaches. Knowledge Mining SFSC185(2005), 255–279 (2005)
11. Pröllochs, N., Feuerriegel, S., Neumann, D.: Generating Domain-Specific Dictionaries using Bayesian Learning. Ecis (2015), 0–14 (2015)
12. Riloff, E., Jones, R., et al.: Learning dictionaries for information extraction by multi-level bootstrapping. In: AAAI/IAAI. pp. 474–479 (1999)
13. Riloff, E., Wiebe, J., Wilson, T.: Learning subjective nouns using extraction pattern bootstrapping. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4. pp. 25–32. CONLL '03, Association for Computational Linguistics, Stroudsburg, PA, USA (2003), https://doi.org/10.3115/1119176.1119180