Self-tuning ongoing terminology extraction retrained on terminology validation decisions

Alfredo Maldonado and David Lewis

ADAPT Centre[⋆], School of Computer Science and Statistics
Trinity College Dublin
Ireland
{alfredo.maldonado,dave.lewis}@adaptcentre.ie

Abstract Automatic terminology extraction (ATE) is a first step in many terminology management processes. When applied on content, a linguistic pattern filter and statistical ranker (a "filter-ranker") produces a ranked list of term candidates to be manually reviewed and validated by a terminologist. Ongoing content creation demands the re-application of ATE on each batch of new content. Unfortunately, traditional filter-rankers cannot learn from previous terminology validation decisions. This paper shows that it is feasible to replace traditional filter-rankers with a machine-learning terminology extraction method that is able to "self-tune" based on terminologists' validations and is thus suitable for ongoing ATE as new content is created. Not only does this method perform better than traditional filter-rankers, but by taking advantage of the manual validation process already in place in terminology extraction workflows, it is possible to improve on similar machine learning systems that are trained only once on a static corpus but are used repeatedly on new content.

Keywords: terminology extraction, ATE, SVM, ACL RD-TEC

1 Introduction

Automatic terminology extraction (ATE) is a first step in the terminology processes associated with many content creation, curation and translation projects. An ATE system is expected to extract a list of specialised, technical or corporate *key* terms from a given corpus or document collection. Terminologists then research, validate, define, manage and/or translate these extracted terms, depending on the actual goal of the ATE effort. The users of the validated and curated glossary (or terminology) are authors, translators, marketing professionals, and other content-creation workers who seek to adhere to organisational, corporate, professional or institutional terminology standards. Style checking and language quality software as well as machine translation systems can also consume this terminology.

Terminology extraction, however, is rarely a once-off affair. As new content gets created, new products, services, features, processes, concepts and other pieces of knowledge get created, all potentially requiring new terminology. Failing to extract terminology at regular intervals along the content creation life-cycle runs the risk of missing the majority of terms by the time the content has grown significantly. This can be shown through the ACL RD-TEC[1] corpus version 1.0 (QasemiZadeh and Handschuh, 2014; see Sec. 4), a collection of Association of Computational Linguistics (ACL) academic papers written between 1965 and 2006, in which specialised terminology has been manually identified and annotated. Figure 1 shows

[⋆] The ADAPT Centre for Digital Content Technology is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund.

[1] http://atmykitchen.info/datasets/acl_rd_tec/. The ACL RD-TEC corpus itself was built upon the ACL ARC corpus (Bird et al., 2008) which was in turn derived from the ACL Anthology http://aclanthology.info.

the proportion of new terms identified in each year's worth of ACL papers and the trend of these proportions across the years. The black dots show the proportion of new terms at each year whilst the blue continuous line depicts the trend, with shaded areas indicating the trend's statistical confidence bounds. For the purposes of this graph, a term is counted as a new term in a given year, if it occurs in at least one paper written in that year and has not been featured in any paper written in any prior year[2]. The proportion of new terms in a given year is the number of new terms in that year divided by the total number of terms in that year.

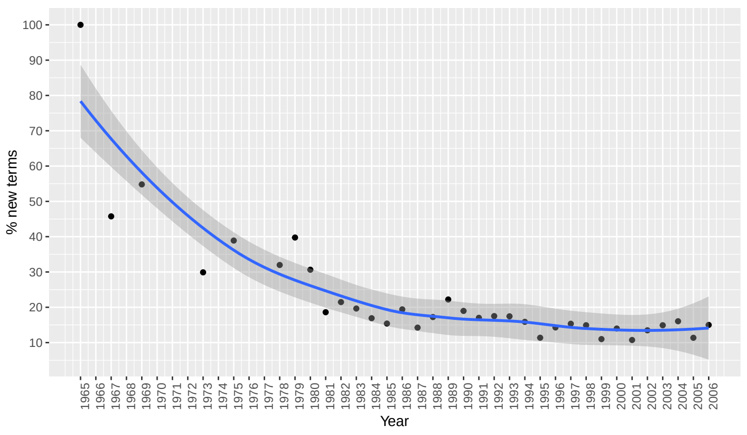

Figure 1. Proportion of new terms per year in the ACL RD-TEC corpus

Notice how the proportion of new terms drops dramatically within the first few years as one would expect. However, the proportion of terms never reaches zero in the long run. Notice that since 1981 the proportion of new terms every year remains relatively constant, fluctuating between 12% and 25%. This implies that failing to do terminology extraction in a subsequent new year will result in missing a substantial amount of new terms. Within a few years, the number of new terms missed will have exceeded the number of terms already captured in all previous years.

As a result, terminology extraction must be conducted repeatedly at appropriate intervals in order to optimally capture the terminology produced by a content-creating organisation, company or professional community. Unfortunately, the standard terminology extraction tools, usually a combination of linguistic pattern filters and statistical rankers ("filter-rankers"), do not readily lend themselves to ongoing use since they will output old or previously extracted terms mixed with new terms. One solution is to automatically filter out previously extracted and/or processed terminology in subsequent runs of a traditional filter-ranker, as proposed by Warburton (2013). However, the filter-ranker will still not be able to generalise from previous validation exercises in order to improve on the quality of newly extracted/ranked terms. In other words, there is no learning (or retraining) feedback loop across successive ATE iterations.

Since terminologists usually review (and validate) the output of terminology extractors, this paper proposes to use the reviewed/validated list of term candidates as feedback to a machine learning terminology extractor (MLTE). The manual effort dedicated during a terminology

[2] As 1965 is the first year in the corpus, 100% of the terms occurring in this year are considered new. There are some years missing a dot (1966, 1968, etc.) This is because no ACL conferences were held in those years.

extraction iteration can thus be used to improve the extractor's performance in the next iteration. By performing simulations of terminology extraction-validation-retraining cycles, this paper shows (Sec. 5) that an MLTE trained in this manner outperforms MLTEs trained only once on a static corpus, as well as traditional filter-rankers.

This paper is structured as follows. Section 2 surveys previous work on ATE, especially on filter-rankers and methods based on machine-learning. It also describes the relationship between our work and Warburton's (2013) proposal. Section 3 presents the general methodology for an MLTE system exploiting the terminology extraction-validation-retraining cycle whilst Section 4 describes how this methodology was adapted for the simulation experiments discussed in Section 5. These simulation experiments were conducted using a subset of the terminology-annotated ACL RD-TEC corpus. As the papers in this corpus are time-stamped, it is possible to group papers in chronological batches. In our experiments, terms are extracted from each chronological batch and compared with the valid terms occurring in the papers belonging to that batch as per the corpus' own terminology annotation, thus simulating the manual validation process. Section 5 also offers conclusions and presents avenues for future research.

2 Previous work

The *de facto* standard method for ATE is a two-step approach. The first step, called the linguistic filtering step, involves identifying sub-sentential fragments from text that satisfy a syntactic pattern from a list curated by a terminologist, domain expert or computational linguist (Nakagawa, 2000; Pazienza et al., 2005). Typical patterns are noun+noun as in *terminology extraction*, adj+noun+noun as in *statistical machine translation*, among others. The second step, the statistical ranking step, ranks the candidates identified in the first step according to some statistical score acting as a proxy for their *termhood* and *unithood* in the text (Kageura and Umino, 1996). Statistical scores include mutual information (Church and Hanks, 1990), log-likelihood ratio of association (Dunning, 1993; McInnes, 2004), Pearson's Chi-Squared test, Student's t-score test, Fisher's exact test (Pedersen, 1996; Purandare, 2004, pp. 35-48), TF-IDF (Spärck Jones, 1972), the C-Value/N-Value/NC-Value methods (Frantzi and Ananiadou, 1996; Frantzi et al., 2000) and even raw term frequency.

There have also been contrastive methods that compare the distribution of a term candidate in a domain-specific corpus against its distribution in a general-language corpus or a corpus from another domain. Methods include relative frequency ratios (Damerau, 1993; Ahmad et al., 1999), domain consensus, domain relevance (Navigli and Velardi, 2002), lexical cohesion of multi-word term candidates[3] (Park et al., 2002; Sclano and Velardi, 2007), etc. In addition, researchers have created composite rankers based on linear combinations of several statistical features, reporting performance improvements over single-feature rankers (Loukachevitch, 2012; Bolshakova et al., 2013).

A third step, often not explored in detail in the ATE literature, consists in manually reviewing the ranked list of term candidates, usually concentrating on the top n ranking candidates or those scoring above some threshold. A subset of these term candidates deemed to be valid terms is then further processed in the terminology pipeline. For the purposes of this paper, this step shall be called the terminology validation step and is conducted by a terminologist, a person who is either a professional terminologist or a trained person with sufficient domain knowledge and linguistic experience to carry out this task competently. Not many works aim to take advantage of this manual validation step with the view of improving future term extraction-validation exercises, despite the development of numerous supervised machine learning approaches that depend on manually-annotated datasets, such as Pecina and

[3] In this work, the lexical cohesion of multi-word term candidates is called *term cohesion*.

Schlesinger (2006), Pecina (2010) and QasemiZadeh et al. (2012), who built linear classifiers based on either logistic regression or support-vector machines (SVMs). A notable exception is Warburton (2013), who proposes to curate exclusion lists of several kinds such as a general lexicon list, a list of already known valid terms, a list of noisy items, etc. These lists are created by a terminologist from the terms automatically extracted using a filter-ranker during the validation of such terms. A computer program keeps track of the validation decisions for each candidate made by the terminologist and thus adds each candidate to the appropriate list. In a subsequent extraction-validation cycle, the terminologist can use these exclusion lists to automatically filter out term candidates produced by the filter-ranker, thus considerably reducing the amount of manual work associated with the validation step. The present paper seeks to combine Warburton's approach with the machine learning classification approach. Instead of curating exclusion lists, we only ask the terminologist to label each extracted candidate as either a valid or a non-valid term via some user interface. The set of labelled term candidates is then used as training data for an SVM classifier.

3 Methodology

The self-tuning ongoing MLTE method sketched at the end of Section 2 can be described more formally as follows. A set of term candidate n-grams $C_t = \{c_1, \ldots, c_n\}$ are extracted and counted from a given batch (set) of documents $b_t = \{d_1, \ldots, d_m\}$ available at some point in time t. For each extracted n-gram $c_i \in C_t$, a record of its part-of-speech (POS) pattern as returned from some parser[4] is also kept: $POS_t = \{pos(c_i), \ldots, pos(c_n)\}$. Given a classifier $f(V_p)$ trained on a set of past validations $V_p = V_{t-k} \cup \cdots \cup V_{t-1}$ from the last k recent batches b_{t-k} to b_{t-1}, each term candidate n-gram c_i is predicted to be a valid term or not by $y_i = f(V_p; c_i)$, where each y_i is a binary label indicating whether term candidate c_i is a valid term (1) or not (0). Those candidates selected (i.e. predicted to be valid) by the classifier are then presented to a terminologist who based on his/her expertise will either confirm or change the validity status of these candidates. This is the so-called validation step. The values confirmed and changed by the terminologist constitute the validations V_t for the current batch b_t, which will be used, along with other recent validations $V_n = V_{t-k+1} \cup \cdots \cup V_t$ to train a new classifier $f(V_n)$ to be used to select term candidates from a batch b_{t+1} to be available at a future time $t + 1$. The collected POS patterns are used to filter out term candidates automatically[5]. Those c_i candidates that have a POS pattern $pos(c_i)$ not shared with at least one valid term from the recent batches b_{t-k} to b_{t-1} are automatically excluded before prediction.

Notice that this method does not require to curate term lists and does not require that terminologists craft any POS pattern filters *a priori*. Notice as well that the number of recent batches to consider, k, for training the classifier makes old data expire automatically. In addition, the method still allows the usage of supplementary filters, for example, excluding terms that are present in the current batch but that are already captured in the terminology database.

4 Experimental setup

4.1 Extraction-validation simulation

The terminology extraction-validation methodology presented in Section 3 is simulated by dividing a subset of the ACL RD-TEC corpus in separate chronological batches following

[4] In this work we use the Stanford Parser (Chen and Manning, 2014) which is applied to full sentences where each candidate n-gram appears.

[5] Whilst this filtering is optional, it is recommended as data points will be highly skewed towards the non-valid class ($y = 0$). This filtering reduces this skewness somewhat.

the corpus' own time-stamping encoded in the paper filenames. Such paper filenames include `C04-1001_cln.xml`, `J04-1003_cln.xml` and `P04-1027_cln.xml`, where the first letter is a code for the ACL journal or conference (i.e. "the venue") where the paper was published and the following two digits indicate the year of publication (2004 in these cases). This is a convention established in the original ACL Anthology. The subset used in the simulation experiments are the 2,781 papers published from 2004 to 2006, gathering a total of 9,114,767 words with each paper tallying an average of 3,300 words. To better simulate extraction-validation iterations conducted at regular intervals, these papers are divided in chronological batches of comparable sizes. Each batch contains at most 40 papers from a single year and venue. On average each batch contains 36.6 papers. This yields a total of 69 separate batches.

The simulation is started by extracting all n-grams ($1 \leq n \leq 7$) from batch b_1 papers. These n-grams are then matched against the valid term[6] annotation in ACL RD-TEC. Non-valid term n-grams that do not share a POS pattern with at least one valid term are automatically filtered out. This set of valid terms and retained non-valid terms constitute the training set for the SVM classifier (actual features described in Sec. 4.2). Then, n-grams are extracted from batch b_2 papers. Those b_2 n-grams that also occurred in b_1 are automatically filtered out. Those b_2 n-grams that do not share a POS pattern with b_1 valid terms are also removed. The remaining set of b_2 n-grams is the test set for the classifier. The classifier trained on b_1 data is then used to predict the validity of each term in the b_2 test set. Evaluation is done by comparing the annotation of each term in the test set with its predicted value (Sec. 4.4). This process is repeated for b_3, except the training data used includes terms from b_1 and b_2. In general, the training data for batch b_t will be the union of terms from batches b_{t-k} to b_{t-1}, where k is the history size, the number of past batches we want to consider as training data. It should be pointed out that regardless of the value of k, terms that occurred in all previous batches (from b_1 to b_{t-1}) are always excluded from b_t's test set as we want to extract new terms only.

4.2 Features

Combinations of several of the statistical features presented in Sec. 2 were explored in preliminary experiments. Based on these, the features selected for the final experiments are two formal binary features, **POS pattern** and **character 3-gram**[7], aimed at making the classifier sensitive to the typical syntactic patterns and morphological shapes of valid terms, as well as two sets of domain contrastive features aimed at detecting terms that are typical of the specialised domain of interest (computational linguistics in this case) and atypical of other (contrastive) domains. These domain contrastive features are:

– **Domain relevance (DR)** (Navigli and Velardi, 2002) measures the degree to which a term t is relevant to domain D_i. It is defined by (1) where $P(t|D_i)$ is estimated by (2).

$$DR(t, D_i) = \frac{P(t|D_i)}{\sum_{j=1}^{n} P(t|D_j)} \quad (1) \qquad P(t|D_i) = \frac{f(t, D_i)}{\sum_{s \in D_i} f(s, D_i)} \quad (2)$$

– **Term cohesion (TC)** (Park et al., 2002; Sclano and Velardi, 2007) seeks to measure the degree of cohesion of multi-word term candidate t in a domain D_i:

$$LC(t, D_i) = \frac{l(t) f(t, D_i) \log f(t, D_i)}{\sum_{w_j \in t} f(w_j, D_i)} \quad (3)$$

where $l(t)$ is the length (number of words) of term candidate t, w_j are the individual words making up t and $f(x, D_i)$ is the frequency of word or term x in domain D_i.

[6] ACL RD-TEC distinguishes between technology and non-technology valid terms. This distinction is not made here.

[7] Notice these character 3-grams are only used as features for the classifier. The extracted term candidates are word n-grams of sizes $1 \leq n \leq 7$. This word n-gram size is a parameter set by the user.

In practice, D_i is some corpus belonging to a particular domain. In this work we model domains from two sources. One is a 2009 dump of Wikipedia[8] clustered into 500 clusters using CLUTO (Karypis, 2003). Each cluster is interpreted to be a topical domain of Wikipedia from which DR and TC scores are computed for a term candidate, yielding two real-valued subvectors of 500 dimensions each. The other source is the history of batches up to the current batch (b_1 to b_t) from the same ACL RD-TEC sample, which is also clustered using CLUTO into c clusters, where c is 2.5% of the number of papers in the history. For each of these clusters DR and TC scores are also computed, yielding another set of real-valued subvectors of c dimensions each.

For each term candidate, the different feature subvectors computed (POS patterns, character 3-grams, Wikipedia and batch-history DR and TC) are concatenated to form a single vector, which is then L^2-normalised.

4.3 Experiments

Three types of experiments are conducted: two baselines and our approach.

– **Baseline 1: Standard single-feature filter-rankers.** We extract term candidates from each batch b_i using the rankers implemented in the JATE Toolkit[9] (Zhang et al., 2008), using n-gram extraction. This baseline follows the same protocol described in Sec. 4.1, except that instead of using a classifier, we use a single-feature statistical ranker. The filtering described in that simulation is also taking place in this baseline. So, essentially, this baseline simulates the process suggested by Warburton (2013) by keeping exclusion lists of previously extracted valid terms and terms with a POS pattern not associated with valid terms.
– **Baseline 2: SVM trained on first batch.** We train the SVM classifier on the first batch and use that classifier to extract terms for all subsequent batches. This baseline simulates the case of an extractor trained once and re-used in all subsequent batches with no retraining.
– **Our approach: SVM trained on last *k* batches.** We conduct the extraction-validation simulation as described in Sec. 4.1 while trying different history sizes k exhaustively.

The SVM experiments employ the LIBLINEAR SVM classifier (Fan et al., 2008).

4.4 Evaluation

Performance is assessed using **precision** and **recall**, which are the standard measures of classifier performance evaluation:

$$P = \frac{valid\ terms\ selected}{total\ \textbf{selected}\ terms} \times 100\% \qquad (4)$$

$$R = \frac{valid\ terms\ selected}{total\ \textbf{valid}\ terms} \times 100\% \qquad (5)$$

A term is deemed to be selected (or extracted) if it is predicted to be a valid term by the classifier. Recall measures the coverage of our extractor (i.e. the percentage of valid terms in a batch that were selected). A low recall value indicates that the extractor is missing many terms. Precision on the other hand measures the percentage of true valid terms from the selected terms. A low precision value indicates that our extractor has produced many false positives. In ATE, we want to identify as many true valid terms as possible, potentially at the risk of having a relatively high number of false positives. So we are interested in achieving high recall at the expense of a moderate precision.

[8] Wikipedia dump kindly cleaned, pre-processed and made available by the Web as a Corpus initiative (Baroni et al., 2009) at `http://wacky.sslmit.unibo.it/doku.php?id=corpora`

[9] `https://code.google.com/archive/p/jatetoolkit/` and `https://github.com/ziqizhang/jate`

Filter-rankers, like those used in the Baseline 1 experiments, cannot be evaluated directly by precision and recall. However, by considering the top N ranked candidates as valid term predictions and all other candidates as non-valid term predictions, one effectively converts a ranker into a classifier that can be evaluated using standard precision and recall (Pecina, 2010). The problem then becomes finding an appropriate value N. If a batch test set contains v valid terms a perfect ranker will return all of these valid terms in the first v positions. So, by setting $N = v$ we will achieve a precision and a recall of 100%. In reality however, the ranker will be expected to be less than perfect. A solution could be to set $N = 2v$, giving the ranker twice the chance of finding all of the valid terms. Notice though that while a ranker will still be able to score a maximum of 100% recall under these conditions, it can only expect to obtain a maximum of 50% precision. Results for four rankers implemented by JATE (C-Value, GlossEx, Raw Term Frequency and Weirdness[10]) are presented in Figure 2. The plots on the left show Precision and Recall when using the conversion $N = 2v$ and the plots on the right show the same performance measures but for the $N = 7v$ conversion strategy. In all plots, the x axis represents the batch being evaluated. Trend lines are also included in the plots. Section 5 discusses these results.

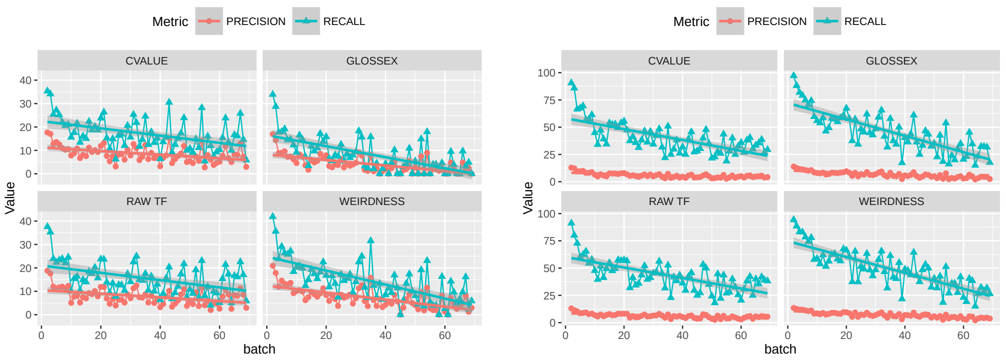

Figure 2. Performance of filter-rankers implemented in JATE (Baseline 1) using $N = 2v$ (left) and $N = 7v$ (right)

5 Results and conclusions

The terminology extraction-validation simulation presented in this paper requires a parameter to be set manually: the history size k, i.e. the number of past batches (b_{t-k} to b_{t-1}) to use as training data for current batch b_t. We conducted systematic experiments trying out each possible history size ($1 \leq k \leq 68$) and found that our method performed quite similarly for most sizes, except for the smallest (1 and 2) which performed slightly worse. We found that on average the best performing history size was 16, which yielded an average recall of 74.17% across all batches. Accordingly, Figure 3 shows results using our method with a history size of 16 (ONGOING) along with the two baselines described in Sec. 4.3 (B1-CVALUE and B2-STATIC). For Baseline 1 only the C-Value ranker is plotted in Figure 3 as this is one of the most widely used rankers.

[10] The Weirdness score (Ahmad et al., 1999) is a relative frequency ratio score closely related to equation (2).

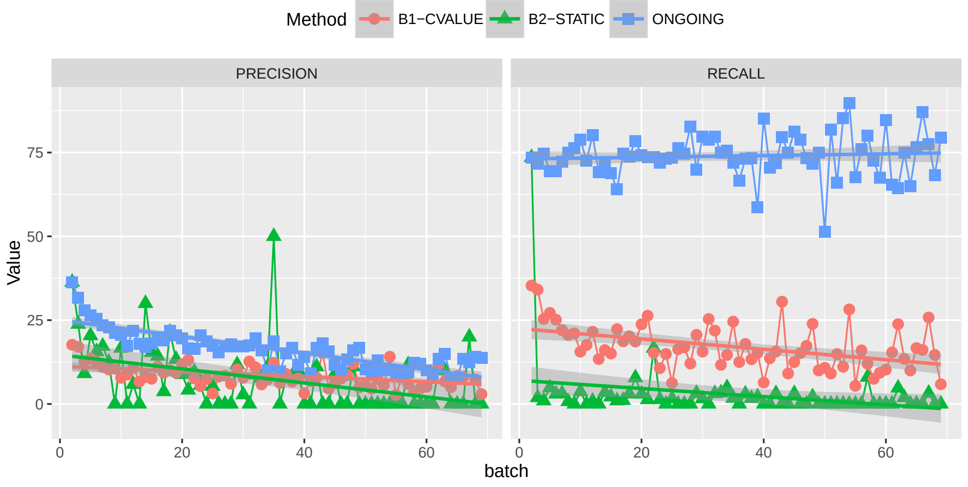

Figure 3. Performance of two baseline methods and our self-tuning ongoing method

The first thing to notice from this figure is that precision scores on the left-hand side tend to be far worse than the recall scores on the right-hand side. This means that all three methods produce a considerable amount of false positives, unfortunately. Our method (ONGOING), however produces the highest recall of all three methods by far, and while fluctuating, the recall trend is slightly on the increase. Compare this performance to that of B2-STATIC, the SVM trained on the first batch and used to extract terms in all subsequent batches with no retraining after each validation. Recall in B2-STATIC performs very well at the beginning (matching ONGOING), but it sinks to the bottom very quickly and largely remains there. This shows that the retraining conducted after each validation is indeed keeping the ONGOING classifier sensitive to new terms. If this retraining stops, the performance drops. So, a terminology extractor trained on a static set of data will not be able to perform well in the long run.

The recall of the C-Value ranker (B1-CVALUE) oscillates between 0 and 30%. Whilst performing better than B2-STATIC, its performance is far more modest than ONGOING, despite the usage of filtering/exclusion lists. In fact, this filtering is perhaps detrimental to the performance of all JATE's filter-rankers. Notice in Figure 2 that both precision and recall decrease without exception as we move towards newer batches. It is possible that the rankers tend to find terms in subsequent batches that are removed by the old term filtering mechanism employed, leaving few good candidates to report. Since the rankers have no information as to the terms being filtered out, they are unable to re-weight the remaining term candidates in order to maintain a high recall level. Whilst filter-rankers can be invaluable in extracting terminology in a new project, they are not suited for ongoing terminology extraction, even when automatic filtering of old terms is implemented, because they lack a feedback loop mechanism.

The results presented in this paper are based on a simulation. In a real terminology validation process, the terminologist has the discretion and flexibility to examine more or fewer term candidates depending on their experience, the quality of terms returned by a classifier or ranker, the size of the batch, among other factors. So, a more realistic, human-based study of a filter-ranker vs. a retrained MLTE is warranted. However, this work does demonstrate that the manual validation process already in place (implicitly or explicitly) in virtually all terminology extraction tasks can be used effectively to retrain machine-learning methods to improve the quality of the extraction itself. In most if not all situations, an extractor based on such a machine-learning method could be readily used as a drop-in replacement for filter-rankers already in place in existing terminology workflows.

For future research we plan to conduct human-based benchmarks in order to confirm the simulations presented here. We also plan to address the low precision scores reported by exploring new features and post-processing strategies like re-ranking the candidates output by the classifier using traditional and new terminology ranking algorithms. Whilst the experiments here focused on one particular dataset from one particular domain (Computational Linguistics academic papers), the method should be applicable to other time-stamped datasets from other domains. So future research will also explore whether classifiers can rely on domain-independent features or whether they must depend on domain-specific features. Finally, the role of the contrastive corpus should be further investigated. Here we employed Wikipedia. However, many, very specific terms will not feature in Wikipedia at all. So we must find a way to cope with those cases. One way would be using a fallback strategy such as relying on sub-terms that do appear in the contrastive corpus. Another would involve using language modelling techniques or distributional MWE composition techniques in order to estimate the values of features of terms missing in the contrastive corpus.

References

Ahmad, K., L. Gillam, and L. Tostevin (1999). University of Surrey participation in TREC8: Weirdness indexing for logical document extrapolation and retrieval (WILDER). In *Proceedings of the Eighth Text Retrieval Conference*.

Baroni, M., S. Bernardini, A. Ferraresi, and E. Zanchetta (2009). The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation 43*(3), 209–226.

Bird, S., R. Dale, B. J. Dorr, B. Gibson, M. T. Joseph, M.-Y. Kan, D. Lee, B. Powley, D. Radev, and Y. F. Tan (2008). The ACL Anthology Reference Corpus: A Reference Dataset for Bibliographic Research in Computational Linguistics. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakesh, pp. 1755–1759.

Bolshakova, E., N. Loukachevitch, and M. Nokel (2013). Topic Models Can Improve Domain Term Extraction. *Advances in Information Retrieval. 35th European Conference on IR Research (ECIC 2013). Lecture Notes in Computer Science. 7814*, 684–687.

Chen, D. and C. D. Manning (2014). A Fast and Accurate Dependency Parser using Neural Networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, pp. 740–750.

Church, K. W. and P. Hanks (1990). Word association norms, mutual information, and lexicography. *Computational Linguistics 16*(1), 22–29.

Damerau, F. J. (1993). Generating and evaluating domain-oriented multi-word terms from texts. *Information Processing and Management 29*(4), 433–447.

Dunning, T. (1993). Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics 19*(1), 61–74.

Fan, R.-E., K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin (2008). LIBLINEAR: A Library for Large Linear Classification. *The Journal of Machine Learning 9*(2008), 1871–1874.

Frantzi, K. T. and S. Ananiadou (1996). Extracting nested collocations. In *Proceedings of the 16th International Conference on Computational Linguistics -Volume 1*, Copenhagen, pp. 41–46.

Frantzi, K. T., S. Ananiadou, and H. Mima (2000). Automatic recognition of multi-word terms:. the C-value/NC-value method. *International Journal on Digital Libraries 3*(2), 115–130.

Kageura, K. and B. Umino (1996). Methods of Automatic Term Recognition: A Review. *Terminology 3*(2).

Karypis, G. (2003). CLUTO - a clustering toolkit. Technical report, Department of Computer Science, University of Minnesota, Minneapolis, MN.

Loukachevitch, N. (2012). Automatic Term Recognition Needs Multiple Evidence. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC)*, Istanbul, pp. 2401–2407.

McInnes, B. T. (2004). *Extending the log likelihood measure to improve collocation identification*. Msc., University of Minnesota.

Nakagawa, H. (2000). Automatic term recognition based on statistics of compound nouns. *Terminology 6*(2), 195–210.

Navigli, R. and P. Velardi (2002). Semantic Interpretation of Terminological Strings. In *Proceedings of the 6th International Conference on Terminology and Knowledge Engineering (TKE 2002)*, Nancy, pp. 95–100.

Park, Y., R. J. Byrd, and B. K. Boguraev (2002). Automatic Glossary Extraction: Beyond Terminology Identification. In *Proceedings of the 19th International Conference on Computational Linguistics*, Taipei.

Pazienza, M. T., M. Pennacchiotti, and F. M. Zanzotto (2005). Terminology extraction: an analysis of linguistic and statistical approaches. *Knowledge Mining 185*, 255–279.

Pecina, P. (2010). Lexical association measures and collocation extraction. *Language Resources and Evaluation 44*(1-2), 137–158.

Pecina, P. and P. Schlesinger (2006). Combining Association Measures for Collocation Extraction. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL 2006)*, Sydney, pp. 651–658.

Pedersen, T. (1996). Fishing for Exactness. In *Proceedings of the South-Central SAS Users Group Conference (SCSUG-96)*, Austin, TX.

Purandare, A. (2004). *Unsupervised Word Sense Discrimination by Clustering Similar Contexts*. Msc thesis, University of Minnesota.

QasemiZadeh, B., P. Buitelaar, T. Chen, and G. Bordea (2012). Semi-Supervised Technical Term Tagging With Minimal User Feedback. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, Istanbul, pp. 617–621.

QasemiZadeh, B. and S. Handschuh (2014). The ACL RD-TEC: A Dataset for Benchmarking Terminology Extraction and Classification in Computational Linguistics. In *Proceedings of the 4th International Workshop on Computational Terminology*, Dublin, pp. 52–63.

Sclano, F. and P. Velardi (2007). TermExtractor: a Web Application to Learn the Common Terminology of Interest Groups and Research Communities. In *Proceedings of the 9th Conference on Terminology and Artificial Intelligence (TIA 2007)*, pp. 8–9.

Spärck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation 28*(1), 11–21.

Warburton, K. (2013). Processing terminology for the translation pipeline. *Terminology 19*(1), 93–111.

Zhang, Z., J. Iria, C. Brewster, and F. Ciravegna (2008). A comparative evaluation of term recognition algorithms. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, pp. 2108–2113.