Transforming Term Extraction: Transformer-Based Approaches to Multilingual Term Extraction Across Domains

Christian Lang[*], Lennart Wachowiak[*], Barbara Heinisch, Dagmar Gromann
University of Vienna, Center for Translation Studies
[christian.lang,lennart.wachowiak]@univie.ac.at

Abstract

Automated Term Extraction (ATE), even though well-investigated, continues to be a challenging task. Approaches conventionally extract terms on corpus or document level and the benefits of neural models still remain underexplored with very few exceptions. We introduce three transformer-based term extraction models operating on sentence level: a language model for token classification, one for sequence classification, and an innovative use of Neural Machine Translation (NMT), which learns to reduce sentences to terms. All three models are trained and tested on the dataset of the ATE challenge TermEval 2020 in English, French, and Dutch across four specialized domains. The two best performing approaches are also evaluated on the ACL RD-TEC 2.0 dataset. Our models outperform previous baselines, one of which is BERT-based, by a substantial margin, with the token-classifier language model performing best.

1 Introduction

Automated Term Extraction (ATE) aims at extracting terms, i.e., single- or multi-word sequences, from domain-specific text. ATE plays a role in many NLP tasks, such as information extraction, knowledge graph learning, and text summarization. In a corpus-level setting, methods range from frequency-based to utilizing Wikipedia links, where no single method has been found to perform consistently best across domains in English (Astrakhantsev, 2018). In document-level ATE, KeyConceptRelatedness (Astrakhantsev, 2014), which relies on keyphrase extraction and semantic relatedness, outperforms other methods (Šajatović et al., 2019). The use of neural networks in these methods is mostly limited to generating embeddings.

A first use of BERT-based language models is documented by Hazem et al. (2020), the winning system of the recent ATE challenge TermEval 2020 (Rigouts Terryn et al., 2020) and the baseline for the proposed approaches. Inspired by this first success of transformer-based models, we compare two variations of the multilingual pretrained language model XLM-RoBERTa (XLM-R) (Conneau et al., 2020) with an innovative use of the multilingual pretrained NMT model mBART (Liu et al., 2020) on the Annotated Corpora for Term Extraction Research (ACTER) dataset (Rigouts Terryn et al., 2019) utilized in TermEval 2020 as well as on the ACL RD-TEC 2.0 dataset (QasemiZadeh and Schumann, 2016). Since masked language and NMT models take sentences as input, the proposed ATE methods operate on sentence level. In spite of this reduced context of sentence input rather than documents or corpora, the models achieve F1 scores of up to 69.8% on ACTER, strongly outperforming the previous baseline of 48.1% .

An XLM-R-based sequence classifier relies on positive (term) and negative (non-term) samples, which are generated based on all n-grams up to a length of six of a given sentence. A second XLM-R-based token classifier decides for each word in a sequence whether it can be considered (part of) a term. Since the second model operates without upfront n-gram generation and only processes each sentence once, it is considerably more time-efficient than the first. Finally, the pretrained NMT model mBART is adapted to transform input sentences to sequences of comma-separated terms, an approach inspired by NMT-based ontology learning (Petrucci et al., 2018).

Analyses of results reveal interesting insights into the performance of the different input processing strategies and transformer-based models, including their ability to handle multi-word terms, training time required, and a comparison between baseline monolingual and multilingual language models in ATE. To achieve sentence-level ATE

[*] Equal contributions

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 3607–3620
August 1–6, 2021. ©2021 Association for Computational Linguistics

the ACTER dataset had to be preprocessed aligning terms with their occurrences in sentences, which we made publicly available together with our source code.[1]

In summary, our main contributions are: (i) We show that transformer-based models can be successfully applied to ATE across three languages and five domains, without the need for text preprocessing or feature extraction; (ii) We show that ATE can be performed successfully on sentence level; (iii) We conduct robust experiments to show that our models outperform competitive baselines; (iv) We investigate the models' abilities to handle single- and multi-word terms, distinct term types, and differences in performance depending on train and test language combinations.

2 Related Work

An initial classification of ATE methods into statistical, linguistic or hybrid (e.g. by Kageura and Umino (1996)) has recently been refined by Astrakhantsev (2018) to methods based on term occurrence frequencies (e.g. C/NC-value Frantzi et al., 2000), occurrence contexts (e.g. Bordea et al., 2013), domain-specific corpora combined with general language corpora (e.g. Weirdness (Ahmad et al., 1999)), topic modeling (e.g. Li et al., 2013), and those utilizing Wikipedia. Methods are additionally categorized by the type of context, i.e., corpus-level (e.g. Zhang et al., 2008; Astrakhantsev, 2018) and document-level (e.g. Šajatović et al., 2019) settings.

These classifications cannot easily accommodate recent neural ATE methods that generally operate on sentence level. An approach most closely related and our baseline by Hazem et al. (2020) utilized RoBERTa (Liu et al., 2019) for English and CamemBERT (Martin et al., 2020) for French and won the TermEval 2020 challenge. In their work, pretrained language models clearly outperformed a classification method based on a variety of features, such as statistical descriptors and the domain-specificity measure termhood (Kageura and Umino, 1996). A recently published approach (Rokas et al., 2020) relies on LSTM, GRU and BERT embeddings and achieves high F1 scores for ATE of Lithuanian terms in the cybersecurity domain. Several approaches build on word embeddings to perform ATE on specific domains, such as medicine (e.g. Bay et al., 2020), or to separate general-language from domain-specific embeddings (Hätty et al., 2020). In contrast, our models perform ATE on four domains and in three languages utilizing a pretrained language and a pretrained NMT model. Extracting terms is also vital to learning expressive ontologies from text, for which Petrucci et al. (2018) train an NMT model to transform sentences to Description Logic formulas, an idea that inspired our NMT-based ATE model.

3 Language Models and NMT

Neural Language Models, which create contextualized language representations, were responsible for many of the recent improvements in NLP. Such models acquire rich contextualized language representations in a pretraining stage in which they learn to predict a masked word in a sentence, a task for which large amounts of training data are readily available. The thereby learned representations can be reused for various downstream tasks in the so-called fine-tuning stage, where task-specific layers are added on top of the pretrained language model. One of the most popular language models is BERT (Devlin et al., 2019), utilizing the transformer architecture (Vaswani et al., 2017). XLM-R (Conneau et al., 2020) is a multilingual variant of BERT, which was pretrained in 100 languages using 2.5 terabytes of Common Crawl data. Moreover, it makes use of the improved training routine introduced by RoBERTa (Liu et al., 2019).

Despite the widespread use of neural language models for NLP, adoption of such self-supervised pretraining approaches in NMT has only recently started to gain traction. NMT is traditionally performed with sequence-to-sequence encoder-decoder models that generate a target language output sequence based on a source language input sequence. Conventional language models trained on predicting masked words from a sequence, such as BERT, have only recently been incorporated into NMT (Zhu et al., 2020). A very interesting alternative is to pretrain an NMT transformer architecture, as done by Lewis et al. (2020) in form of a Bidirectional and Autoregressive Transformer (BART) (Lewis et al., 2020). This is achieved by combining a bidirectional encoder similar to that of BERT with an autoregressive decoder, as seen in GPT (Radford et al., 2018). Thereby, contextualized language representations are trained and

[1] https://github.com/Text2TCS/Term-Extraction-With-Language-Models and https://github.com/Text2TCS/mBART-termextraction

a model that is proficient in text generation and translation is created. Liu et al. (2020) applied the BART architecture to large-scale monolingual corpora across 25 languages, creating multilingual BART (mBART) that can be directly fine-tuned for machine translation (MT).

4 Dataset

In order to compare to a strong baseline, we train and test on the ACTER dataset (Rigouts Terryn et al., 2019) utilized in the recent TermEval 2020 challenge. The domains wind energy and corruption represent the training set, dressage (equitation) the validation set, and heart failure the hold-out test set, for which the count of words and unique gold standard terms including named entities for English, French and Dutch are presented in Table 1.

In the ACTER dataset, words were labeled as specific, common, and out-of-domain (OOD) terms, and named entities (NE). Specific terms are understood by domain experts, while common terms might also be additionally understood by laypersons. OOD terms might be specific to a different domain, but used in the domain at hand, e.g. statistical terms in the medical domain.

Since the time of the challenge the dataset has undergone some minor updates, that is, unicode encoding, dash and quote normalization.[2] We believe that these minor normalization changes do not significantly impact comparability to TermEval results, which is confirmed by the fact that our most similar model to the baseline, the sequence classifier, achieves comparable results. Furthermore, the ACTER dataset provides terms as a single list for all documents in a domain. However, we required inline sentence-level term annotation, which we generated. In rare cases, this generation of inline annotations might have lead to erroneous results for single-word terms. For instance, the term "gain" as in "private gain" lead to the verb "gain" as in "gain acceptance" to be erroneously annotated in the corruption domain. We manually analyzed 300 inline annotated sentences and since the above example was the only error found, we consider this a negligible issue.

The fully inline annotated dataset ACL RD-TEC 2.0 (henceforth ACLR2) dataset provides cleaner training and test data and could therefore potentially further boost model performance as we show in Section 7.2. The ACLR2 dataset provides a total of 471 inline human annotated abstract texts from articles in the ACL Anthology Reference Corpus. As shown in the split of numbers in Table 1, two separate annotations by two human experts are provided. Since no official train/val/test split is provided, we chose to split the ACLR2 dataset with a 60/20/20 split per annotator. In contrast to the ACTER dataset, ACLR2 is only available in English and exclusively covers scientific abstracts in the domain of computational linguistics. In terms of baseline, previous work generally reported precision at *k* top terms extracted (P@k) (Zhang et al., 2018b) or F1 on Recoverable True Positives (F1@RTP) (Zhang et al., 2018a), due to the necessity to define an arbitrary cut-off point with traditional ATE methods. In another work attempting ATE with neural networks, due to the lack of an official data split and a restriction to domain specific terms, F1 scores are reported on arbitrary parts of the dataset (Kucza et al., 2018).

ACTER	**Train**	**Val**	**Test**
Words_{en}	97,145	51,470	45,788
Terms_{en}	2,708	1,575	2,585
Words_{fr}	106,792	53,316	46,751
Terms_{fr}	2,185	1,183	2,423
Words_{nl}	96,887	50,882	47,888
Terms_{nl}	2,540	1,546	2,257
ACLR2	**Train**	**Val**	**Test**
$\text{Words}_{an.1}$	11,473	3,846	4,032
$\text{Terms}_{an.1}$	1,306	420	477
$\text{Words}_{an.2}$	16,939	5,757	5,441
$\text{Terms}_{an.2}$	1,743	583	673

Table 1: Train, validation, and test split by word count and term count per language/annotator

5 Neural Language Model-based ATE

We introduce two possible architectures for ATE based on the multilingual language model XLM-R. For the experiments we use the base-size model version in form of the implementation made available by the transformers library (Wolf et al., 2019).

5.1 Sequence Classifier

As with the winning approach of TermEval 2020 (Hazem et al., 2020), our first architecture utilizes language models for binary sequence classification by using a fully connected layer to classify the representation of the special classification token <s>,

[2]This normalized version 1.4 is available at https://github.com/AylaRT/ACTER

which encoded by XLM-R carries information regarding the whole input sequence. Instead of using language specific models, however, we make use of the multilingual model XLM-R, which enables the use of a single model for all languages and has the ability to generalize to unseen languages.

The model receives pairs consisting of a term candidate and a context sentence in which the candidate appears as input as exemplified in Table 2. Term candidates are created by producing all possible n-grams of a given sentence. Due to performance reasons and the term length distribution in the dataset (mostly <5 words), n-grams were only created up to a length of 6 words. For instance, given the input sentence "We meta-analyzed mortality using random-effect models" a positive sample, i.e., one labeled as term, is "random-effect models. We meta-analyzed mortality using random-effect models", while a negative sample is "mortality using. We meta-analyzed mortality using random-effect models". For training the model, we undersample the negative samples so that their amount matches the amount of positive samples to compare to Hazem et al. (2020). For the evaluation on the validation and test set we use all possible n-grams for each input sentence, thus, creating a set of extracted terms which we can evaluate against the gold standard. The model was trained for 4 epochs with a batch size of 32 using the Adam optimizer with a learning rate of 2e-5.

5.2 Token Classifier

The second architecture we use for experimentation classifies each token of an input sentence separately, utilizing the same fully connected layer for all tokens after they have been processed by XLM-R. This leads to a significant reduction in training and inference time as each sentence has to be only processed once by XLM-R. This type of architecture is usually utilized in tasks like Named Entity Recognition (NER) (Devlin et al., 2019), where each word of a sequence needs to be classified.

The input provided to the model now simply consists of the sentences of the document which we want to process. The model then assigns each input token one of three possible output labels: "B-T" for the beginning of a term, "T" for the continuation of a term, and "n" in the case the token is not part of a term. For instance, the input sentence "We meta-analyzed mortality using random-effect models." would be labeled as 'n', 'B-T', 'B-T', 'n', 'B-T', 'T', 'n', with the last label annotating the punctuation at the end of the sentence.[3] Table 2 compares this input and output pattern with the other two methods. Since XLM-R's tokenizer is a SentencePiece tokenizer that splits the input into tokens on a subword level, the output labels obtained from the model are also subwords and have to be matched to the original words of the sentence afterwards. For training we used the Adam optimizer with a learning rate of 2e-5. Moreover, we used a batch size of 8 evaluating the model every 100 steps to be able to load the best model at the end.

6 NMT-based ATE

As a third experiment, we present a novel approach to ATE building on a recent sequence-to-sequence denoising auto-encoder model trained for NMT. We chose the recent and robust mBART model trained on the Common Crawl corpus in 25 languages (mBART25) (Liu et al., 2020) available in the Fairseq library (Ott et al., 2019).

6.1 Data Preprocessing for NMT-based ATE

Since we construct the downstream task of ATE as an MT task, we required parallel text data for supervised fine-tuning of mBART. We opted for a sentence-level approach, which specifically requires sentence-aligned parallel data. Sentence tokenization was performed with the Punkt tokenizer of NLTK and terms were inline annotated with the flashtext algorithm (Singh, 2017). For the ACLR2 dataset, individual sentences and the terms within were extracted with an XML parser. In order to distinguish single- and multi-word terms in the model's output sequence, a separator between terms or a unifying character between components of multi-word terms was required. Preliminary testing showed that using a semicolon surrounded by white-spaces (;) as separator would achieve the same final F1 score as using more complex separators like a tag (for example <term>). Notably using an underscore (w_w) to connect the individual constituents of a term (w) lowered the score of the output significantly, that is, F1 performance of the best model was 5.3% lower on average across all test languages when compared to utilizing semicolons. Irrespective of the separator, the model would at times add or omit a white-space between separator and term, which had the effect that the

[3] The separation of "meta-analyzed" and "mortality" as distinct terms corresponds to the gold standard.

Model	Input Example	Output Example
Sequence Classifier	random-effect models. We meta-analyzed mortality using random-effect models	Term
Token Classifier	We meta-analyzed mortality using random-effect models	['n', 'B-T', 'B-T', 'n', 'B-T', 'T', 'n']
NMT	We meta-analyzed mortality using random-effect models	meta-analyzed ; mortality ; random-effects models

Table 2: Input and output examples for all three transformer-based models

term would not be considered in the evaluation. This was remedied in the process of extracting individual terms from the output sequence and the results reported in Section 7 are with unwanted white-spaces removed. Tokenization during training was performed with SentencePiece (Kudo and Richardson, 2018) and data was binarized with the *fairseq-preprocess* CLI tool.

6.2 NMT Model Fine-Tuning

The pretrained mBART model was fine-tuned with the preprocessed data described in Section 6.1. Input to the encoder model was a given sentence, such as "Codes of conduct forbid corruption, irrespective of its intended purpose.", while the decoder would be shown the expected term labels, such as "codes of conduct ; corruption". No language-specific tags were added to input or output, which is compared to the other methods in Table 2. For faster and more memory-efficient training we used automated mixed precision training of Fairseq with the Fused Adam Optimizer of the NVIDIA Apex PyTorch extensions.[4] We fine-tuned a separate model for each language of the dataset and a single model with all languages combined. Following the original publication of the pretrained model, each model was fine-tuned with 0.3 dropout, 0.2 label smoothing, 2500 warm-up steps and a learning rate of 3e-5. Furthermore, we opted for a dynamic batch size by limiting the maximum tokens per batch to 768, while updating the gradients every 4 steps (more details in Section 7.4).

While preliminary testing showed faster convergence and slightly higher final scores with higher tokens per batch, availability of the V100 GPU was not guaranteed and therefore training hyperparameters had to be adjusted to also run on an RTX2080Ti GPU, which limited the maximum tokens per GPU to 768. Model performance was evaluated every full epoch. Results were generated using the standard generation parameters of Fairseq.

[4] https://github.com/NVIDIA/apex

7 Results

This section first presents the results on ACTER including an analysis per language (combination) and the results on ACLR2, then details the term length and type behavior of the models, and finally compares their training time efficiency. We additionally report on the validation performance of the best performing token classifier in Table 4, which shows some performance differences to the test domain, especially with French as training and validation language. For further comparability we also provide precision, recall and F1 scores at k top terms of 15 methods offered by the term extraction toolkit ATR4S, which implements a large range of existing ATE methods, in Appendix A.

7.1 Results on ACTER

To compare our results to the strongest participant of TermEval 2020, we report precision, recall and F1 scores in Table 3. These metrics are calculated on the basis of the available annotation in the original ACTER dataset, where we opted for the more comprehensive list of terms including named entities. All three models are evaluated on different combinations of training and test languages as shown in Table 3, where the heart failure domain is the hold-out test set as done for the SOTA baseline. The overall best results are marked in bold for each test language, while the best results of each model (if not bold) are highlighted in italics.

The overall best result for our approaches was an F1 score of 69.8%, which could be achieved by training the token classifier model on English and testing it on Dutch. With the exact same settings as the baseline (Hazem et al., 2020) that is based on RoBERTa (Liu et al., 2019) for English as training and test language, the token classifier

Training	Test	Sequence Classifier			Token Classifier			NMT			Previous SOTA		
		Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
EN	EN	30.9	84.0	45.2	54.9	62.2	58.3	45.7	63.5	53.2	34.8	70.9	46.7
FR	EN	31.1	79.5	44.7	56.7	36.2	44.2	50.0	59.3	54.2			
NL	EN	22.3	91.1	35.9	55.3	61.8	**58.3**	48.3	64.3	55.2			
ALL	EN	31.4	85.8	*46.0*	54.4	58.2	56.2	50.2	61.6	*55.3*			
EN	FR	34.6	79.0	*48.1*	65.4	51.4	**57.6**	48.8	61.3	54.4			
FR	FR	32.2	80.2	46.0	68.7	43.0	52.9	52.7	59.6	55.9	44.2	51.5	48.1
NL	FR	26.1	84.7	40.0	62.3	48.5	54.5	54.3	60.9	57.4			
ALL	FR	33.2	78.9	46.7	62.7	49.4	55.3	55.0	60.4	*57.6*			
EN	NL	42.8	89.8	*58.0*	67.9	71.7	**69.8**	48.8	63.9	55.4			
FR	NL	41.3	87.6	56.1	69.2	55.2	61.4	56.2	63.4	59.6			
NL	NL	32.7	94.1	48.5	71.4	67.8	69.6	60.6	70.7	*65.2*	18.9	18.6	18.7
ALL	NL	40.4	91.5	56.0	70.0	65.8	67.8	60.6	70.0	64.9			

Table 3: Test set results represented by training and test languages of the ACTER heart failure domain and in comparison to the state-of-the-art (SOTA) results from TermEval 2020.

Training	EN Val	FR Val	NL Val
EN (ACTER)	55.6	45.3	60.5
FR (ACTER)	41.9	33.6	49.6
NL (ACTER)	54.6	47.7	57.8
ALL (ACTER)	50.0	40.4	51.5
ACLR2 An.1	75.5	/	/
ACLR2 An.2	79.3	/	/

Table 4: Validation performance of token classifier on dressage domain of the ACTER dataset and 20% validation data of the ACLR2 dataset.

achieves an 11.6% higher F1 score and the NMT model an improvement of 6.5% on the F1 score. The sequence classifier struggles with precision and cannot outperform the baseline in this setting. Best performance for English as test language can be achieved by the token classifier trained on Dutch and by the NMT model trained on all languages.

When testing on French, the sequence classifier is on par with the F1 baseline (Hazem et al., 2020) building on CamemBERT (Martin et al., 2020), while the token classifier outperforms it by 9.5% and the NMT model obtains an additional 7.8%. Best performance on French as a test language is achieved by the token classifier when trained on English and by the NMT model when trained on all languages again. The baseline for Dutch is provided by a bidirectional LSTM with GLOVE.[5] With Dutch as a test language, the sequence and token classifier achieve their best result when trained on English, the NMT model when trained on Dutch.

A significant result is the substantial improvement of precision of the token classifier and NMT model over the baseline, even though the recall for English as test language lags behind. For French, the recall could be improved with the NMT model and matched by the token classifier when trained on English. Interestingly, the sequence classifier achieves a remarkable improvement on recall, however, lags behind on precision for all settings.

This can be explained by the fact that we perform undersampling of the negative samples to match the number of positive samples, a strategy adopted from Hazem et al. (2020) to obtain comparable results. If undersampling is reduced, the precision and recall scores are more balanced and closer to the performance of the token classifier, however, training time is considerably increased. Another reason for the higher number of extracted phrases by the sequence classifier compared to the other models is that it can extract multi-word terms as well as words which are part of these multi-word terms separately, since both are used as input in the form of potential term candidate n-grams.

All three models show remarkable zero-shot transfer learning capabilities, i.e., they are trained on one language and show strong test scores on another. This is especially true for the token classifier, where models trained on a single language often outperform those trained on all three languages. This transfer learning ability across languages can also be observed in the overall highest F1 scores

[5]No system description paper was submitted for this approach after participation in the challenge.

for the English test set, which was achieved by a model trained on Dutch, and for the French test set, which was achieved by a model trained on English.

7.2 Results on ACLR2

In addition to evaluating our models on the ACTER dataset, we compared the two best performing architectures, i.e., the token classifier and the NMT model, on the ACLR2 dataset. Both models achieve similar test scores as reported in Table 5 and higher than the scores achieved on the ACTER dataset. As with the ACTER dataset, we additionally report validation performance of the best performing token classifier model in Table 4, which is in line with the test performance.

Data	Token Classifier			NMT		
	Prec	Rec	F1	Prec	Rec	F1
An.1	74.4	77.2	75.8	73.2	77.2	75.2
An.2	80.1	79.3.	80.0	79.4	80.7	80.0

Table 5: Test set results of token classifier on data from Annotator 1 and 2 of the ACLR2 dataset.

7.3 Term-based Analysis

A qualitative analysis of the lists of false positives and false negatives based on the ACTER dataset demonstrated that all models handle acronyms well. This may be due to the text type in ACTER, which is partially based on scientific abstracts that frequently introduce acronyms in brackets. If acronyms are part of the term, e.g. "LV strain rate", there was a high number of false negatives in both models. Moreover, false negatives occurred in all models if a term included a proper name and an apostrophe, e.g. "Chaga's disease" or "Cronbach's α", or frequently if it included a figure, e.g. "p38alpha" or "6-min walk test". In addition, named entities that included version numbers or consisted of multiple words often resulted in false negatives, e.g. "Self-Care of Heart Failure Index Version 6.2", "Multicenter Automatic Defibrillator Implantation Trial-Cardiac Resynchronization Therapy". In the token classifier and NMT model, the class of named entities of cities, e.g. "New York" and "Seattle", were frequently not identified as terms. False negatives also occurred in all models if it was a particularly long multi-word term, e.g. "resynchronization reverses remodeling in systolic left ventricular dysfunction". A tendency by the token classifier to split longer terms could be observed, e.g. splitting adjectives and nouns.

To quantitatively evaluate how well the different model types handled terms of different lengths, we computed the F1 scores individually for terms of a specific length, based on the terms in the ACTER test set. The results in Table 6 were computed using the best model of each method, i.e., the model trained in English for the token and the sequence classifier and the model trained on all language for the NMT model. We can see that the scores of all models decrease with term length. Secondly, we observer that for English and Dutch the token classifier has the strongest results for all term lengths. However, for French the token classifier scores strongly decrease for multi-word terms, even though it is still the best model for unigrams. This is due to a very low recall, e.g. for 4-grams and higher the token classifier recalls only 7% of all French terms. The NMT model shows more consistency between languages, thus, performing strongest for French multi-word terms. As already the case with the overall scores the sequence classifier shows the highest recall values for both single-word and multi-word terms, however, lagging behind in precision, which leads to an overall lower F1 score.

Furthermore, based on the ACTER term type annotation (see Section 4), we could compare the types of terms extracted by the individual models. As can be seen in Fig. 1, the models all achieve a very similar distribution of extracted term types when compared to the gold test set distribution. We can observe, however, that the sequence classifier showed a slight tendency to extract more common and OOD terms and noticeably less NEs than the other models. All models tended to extract more specific terms, with the token classifier and the NMT model interestingly extracting comparatively few OOD terms.

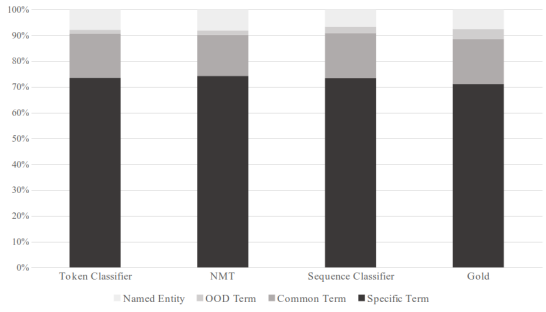

Figure 1: Distribution of term types across languages in the models' true positives and the ACTER gold test dataset.

Term Length	Sequence Classifier (F1)			Token Classifier (F1)			NMT (F1)		
	EN	FR	NL	EN	FR	NL	EN	FR	NL
Unigram	61.6	61.2	68.0	**63.3**	**69.1**	**73.8**	61.7	61.8	70.4
Bigram	38.0	41.0	39.8	**55.9**	43.0	**58.4**	52.2	**57.6**	56.3
Trigram	37.0	35.4	40.4	**55.4**	31.3	**49.1**	51.6	**52.4**	47.7
$\geq$ 4-gram	32.2	22.6	30.3	**44.3**	12.2	**42.9**	43.4	**33.4**	38.8

Table 6: F1 Scores based on different term lengths using the overall best model for each method on the ACTER dataset. In bold the best scores per row for each language.

7.4 Training Time Efficiency

Looking at the epochs required to reach the best score on the ACTER validation set, we can observe that in most cases the token classifier model requires not even a single training epoch. Training with the English dataset required 300 steps with a full epoch consisting of 432 steps. The model trained on French was the only model with its best performance being reached during the second epoch after 700 steps while a full epoch consists of 437 steps. The model trained on Dutch performed best after 400 steps while one epoch takes 553 steps. The multilingual model converged the quickest needing only 200 steps whereas a full epoch consists of 1,421 steps. The token classifier models trained on the ACLR2 dataset need more epochs and achieve their highest scores after 3 and 5 epochs respectively. However, due to the lower training set size of the ACLR2 corpus, this also corresponds to less than 500 steps, thus, being similar with the training times reported for the models trained on the ACTER data. In comparison, the sequence classifier achieved its best performances on the ACTER validation set after 4 epochs of training.

The NMT model also required several epochs to reach the best performance. Initially, all models were trained for 80 epochs, with the model having the lowest validation loss being loaded at the end. The models trained on monolingual data benefited from longer training compared to the models trained on the combined multilingual data. For completeness, we report the training epochs, label smoothed cross entropy loss, and log perplexity on the validation set for the best models. For the English dataset the reported score was achieved at epoch 49 with a loss of 5.82 and perplexity of 3.94. For the French dataset peak performance was reached at epoch 40, with a loss of 5.82 and perplexity of 3.78. Like the French model, the Dutch model achieved its best performance at epoch 40 having a loss of 5.69 and a perplexity of 3.37. When trained on one language, model performance was observed to drop for unseen languages when training beyond the best validation score. For instance, while the English model at epoch 49 obtained F1 scores of 53.2%, 54.4%, 55.4% for the English, French, and Dutch test data respectively, at epoch 80 these scores were at 53.6%, 50.6% and 52.1% respectively, gaining little for English and losing for unseen languages. Finally, for the multilingual dataset the model reached the reported peak performance already at epoch 22 as it trains on a lot more data per single epoch. Loss and perplexity were at 5.50 and 2.89 respectively. The training and validation times as well as the used GPUs are reported in Table 7. Training times denotes the full training time over all epochs without any validations. Validation time denotes the time for a single validation. The token classifier is the most efficient.

Model	Train Time	Val Time	GPU
Seq.	19	44	P100
Tok.	9	1	P100
NMT	49	2	V100

Table 7: Training/validation times in minutes on the English ACTER data and GPUs used.

8 Discussion

Although the ACLR2 dataset is smaller in size than the ACTER dataset, the resulting F1 scores are considerably higher. Apart from the fact that it only covers a single domain, ACLR2 already provides inline annotations and more consistent term annotations, which seems to facilitate learning the task. Inconsistencies in the ACTER annotations were mainly noted when analyzing false positives of the models. For instance, "patient" is considered a common term in the heart-failure domain, but "serum" is not annotated at all, although in our view it would also qualify as common term.

We also noted that more training data does not necessarily increase model performance. As indicated by the training times on the ACTER dataset, the token classifier achieved its best evaluation scores long before training for a whole epoch, i.e., having seen only a small fraction of the available data before reaching its strongest performance.

In this paper we compare the performance of a pretrained monolingual language model baseline with pretrained multilingual language models. Previous work indicates that monolingual language models like RoBERTa or CamemBERT outperform multilingual language models on tasks posed in a single language (Rönnqvist et al., 2019). The difference increases the higher the complexity of the given task but is negligible on simple tasks that mostly rely on syntactic features. Since in our case the multilingual model XLM-R in form of a sequence classifier performs very similar to the sequence classifier-based RoBERTa model winning TermEval 2020, it indicates that successful ATE does not require very strong language understanding but corresponds more to simpler tasks relying mostly on syntactic features. Nevertheless, the remarkable zero-shot transfer learning of the multilingual models fine-tuned on a single language would also suggest that the multilingual pretraining might aid the model in defining what a term is, as highly domain-specific terms might be similar between languages tested, e.g. rooted in Latin. In the NMT output analysis, we found that the knowledge transfer between languages could cause curious side-effects, where at times terms are predicted by the model in a semi-translated way. For instance, when training on English the model would at times invent "toxicity cardiaque " for the French test set instead of extracting "toxicité cardiaque".

Besides stronger performance, the NMT model as well as the token classifier have a higher potential to better handle the possible extension of the term extraction task to include discontinuous entities, which, however, are so far not annotated in the datasets we used. An example of a discontinuous entity can be found in the expression "left and right ventricular failure", where "right ventricular failure" but also "left ventricular failure" are terms, the latter not being continuous in the original expression. While the NMT model does not require any special adaptations to deal with such an addition, the sequence classifier would have to consider many more n-gram combinations leading again to even higher training and inference times per sentence. To consider discontinuous entities with the token classifiers labels, the annotation and training process would have to be adapted to a multi-label token classification, e.g. the above phrase would be labeled as [B-T, n, n, T, T] and [n, n, B-T, T, T]. Since in the first label "ventricular" and "failure" are labeled as "T" they still clearly belong to the word "left" labeled as "B-T", which could be considered in a post processing step.

9 Conclusion

In this paper, we adapt and evaluate three transformer-based models on the task of ATE, building on pretrained multilingual language and NMT models. In this evaluation, these multilingual models outperform a baseline of monolingual language models and show remarkable zero-shot abilities. A token classification strategy building on a language model achieved the best performance, however, the NMT-based model seemed to be able to handle multi-word expressions more consistently across languages and not lag far behind in performance. One aspect that became very clear is a prevalence for quality over quantity when fine-tuning pretrained models to the task of ATE.

Recently, both NMT and masked language models show a trend towards increased input sequence capacity. Thus, it would be interesting to evaluate the impact of context length on the proposed models by testing with more domain context than only single sentences. Furthermore, to test the ability of the token classifier and the NMT model to handle discontinuous terms, such as elliptical expressions, a dataset containing and annotating such terms would be interesting.

Acknowledgments

The Text2TCS project[6] was supported by the European Language Grid project through its open call for pilot projects. The European Language Grid project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement no. 825627 (ELG). The computational results presented have been achieved [in part] using the Vienna Scientific Cluster (VSC).

[6] https://text2tcs.univie.ac.at/

Impact Statement

Automatically extracting domain-specific terms across domains and languages with high accuracy provides a valuable means to reduce time and resource effort in creating terminological resources. Such resources are important to ensure terminological consistency in specialized communication, such as communication between different groups in times of crisis, and to avoid misunderstandings.

From a technological perspective, we introduce multilingual pretrained language models to the field of Automated Term Extraction (ATE) with detailed tests on three different transformer-based models across four domains and three languages. Since these models support considerably more languages than tested, the approach can be transferred to other languages. This transfer capability has been tested by training in a specific language and then testing models in another language. Transfer capabilities extend to domains, since we trained and validated on three domains and achieved results strongly outperforming previous approaches on a previously unseen test domain. Up to this point, such flexibility has been achieved by statistical approaches, however, with considerably lower results in precision and recall. In contrast to previous ATE methods performing on corpora, our models extract terms on sentence level. This makes ATE more flexible since neither large domain-specific nor reference corpora are required.

From a societal perspective, terminological inconsistencies are a major source of misunderstanding in the communication among experts, between experts and laypersons, and between laypersons in reference to a specialized domain. This issue can be mitigated by publishing agreed upon designations for real-world phenomena in a specialized domain that can be consulted for domain-specific communication. However, manually preparing a collection of natural language terms is extremely human resource- and time-intensive. We reduce this workload for governmental institutions, private and public organizations, and private persons by providing a method to automate the detection of such domain-specific terms in natural language texts across languages and domains.

In terms of risk, such a highly flexible solution to automated term extraction fully depends on the quality of the input text. Misleading, erroneous, or biased contents will inevitably be propagated to the resulting terminologies. Relying on terminologies extracted from such problematic contents can negatively impact specialized communication or conclusions drawn from it. Thus, it is of vital importance for any user of this approach to mitigate the uncertainty of the reliability of extracted terms by only considering high-quality and reliable sources in the term extraction process and have domain experts carefully review the outcome prior to utilizing it in communication. We cannot guarantee that in a real-life setting all important terms have been extracted and all extracted terms are indeed central to the domain at hand. Furthermore, training neural network models is a process known to leave an environmental footprint, which we try to mitigate by fine-tuning pretrained models. Fine-tuning is less resource- and time-intensive than training from scratch, but still requires high-performance computing clusters.

References

Khurshid Ahmad, Lee Gillam, Lena Tostevin, et al. 1999. University of surrey participation in trec8: Weirdness indexing for logical document extrapolation and retrieval (wilder). In *TREC*, pages 1–8.

Nikita Astrakhantsev. 2014. Automatic term acquisition from domain-specific text collection by using wikipedia. *Proceedings of the Institute for System Programming*, 26(4):7–20.

Nikita Astrakhantsev. 2018. ATR4S: toolkit with state-of-the-art automatic terms recognition methods in scala. *Language Resources and Evaluation*, 52(3):853–872.

Matthias Bay, Daniel Bruneß, Miriam Herold, Christian Schulze, Michael Guckert, and Mirjam Minor. 2020. Term extraction from medical documents using word embeddings. In *4th IEEE Conference on Machine Learning and Natural Language Processing (MNLP 2020)*. IEEE Computer Society.

Georgeta Bordea, Paul Buitelaar, and Tamara Polajnar. 2013. Domain-independent term extraction through domain modelling. In *The 10th international conference on terminology and artificial intelligence (TIA 2013), Paris, France*. 10th International Conference on Terminology and Artificial Intelligence.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Katerina Frantzi, Sophia Ananiadou, and Hideki Mima. 2000. Automatic recognition of multi-word terms:. the c-value/nc-value method. *International journal on digital libraries*, 3(2):115–130.

Anna Hätty, Dominik Schlechtweg, Michael Dorna, and Sabine Schulte im Walde. 2020. Predicting degrees of technicality in automatic terminology extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2883–2889, Online. Association for Computational Linguistics.

Amir Hazem, Mérieme Bouhandi, Florian Boudin, and Beatrice Daille. 2020. TermEval 2020: TALN-LS2N system for automatic term extraction. In *Proceedings of the 6th International Workshop on Computational Terminology*, pages 95–100, Marseille, France. European Language Resources Association.

Kyo Kageura and Bin Umino. 1996. Methods of automatic term recognition: A review. *Terminology. International Journal of Theoretical and Applied Issues in Specialized Communication*, 3(2):259–289.

Maren Kucza, Jan Niehues, Thomas Zenkel, Alex Waibel, and Sebastian Stüker. 2018. Term extraction via neural sequence labeling a comparative evaluation of strategies using recurrent neural networks. In *Proceedings of INTERSPEECH 2018*, pages 2072–2076.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Sujian Li, Jiwei Li, Tao Song, Wenjie Li, and Baobao Chang. 2013. A novel topic model for automatic term extraction. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 885–888.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Giulio Petrucci, Marco Rospocher, and Chiara Ghidini. 2018. Expressive ontology learning as neural machine translation. *Journal of Web Semantics*, 52:66–82.

Behrang QasemiZadeh and Anne-Kathrin Schumann. 2016. The ACL RD-TEC 2.0: A language resource for evaluating term extraction and entity recognition methods. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1862–1868, Portorož, Slovenia. European Language Resources Association (ELRA).

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Ayla Rigouts Terryn, Veronique Hoste, Patrick Drouin, and Els Lefever. 2020. TermEval 2020: Shared task on automatic term extraction using the annotated corpora for term extraction research (ACTER) dataset. In *Proceedings of the 6th International Workshop on Computational Terminology*, pages 85–94, Marseille, France. European Language Resources Association.

Ayla Rigouts Terryn, Véronique Hoste, and Els Lefever. 2019. In no uncertain terms: a dataset for monolingual and multilingual automatic term extraction from comparable corpora. *Language Resources and Evaluation*, 54:385–418.

Aivaras Rokas, Sigita Rackevičienė, and Andrius Utka. 2020. Automatic extraction of lithuanian cybersecurity terms using deep learning approaches. In *Human Language Technologies–The Baltic Perspective*, volume 328, pages 39–46. IOS Press.

Samuel Rönnqvist, Jenna Kanerva, Tapio Salakoski, and Filip Ginter. 2019. Is multilingual BERT fluent in language generation? In *Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing*, pages 29–36, Turku, Finland. Linköping University Electronic Press.

Antonio Šajatović, Maja Buljan, Jan Šnajder, and Bojana Dalbelo Bašić. 2019. Evaluating automatic term extraction methods on individual documents. In *Proceedings of the Joint Workshop on Multiword Expressions and WordNet (MWE-WN 2019)*, pages 149–154, Florence, Italy. Association for Computational Linguistics.

Vikash Singh. 2017. Replace or retrieve keywords in documents at scale. *CoRR*, abs/1711.00046.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Ziqi Zhang, Jie Gao, and Fabio Ciravegna. 2018a. Semre-rank: Improving automatic term extraction by incorporating semantic relatedness with personalised pagerank. *ACM Trans. Knowl. Discov. Data*, 12(5).

Ziqi Zhang, Jose Iria, Christopher Brewster, and Fabio Ciravegna. 2008. A comparative evaluation of term recognition algorithms. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Ziqi Zhang, Johann Petrak, and Diana Maynard. 2018b. Adapted textrank for term extraction: A generic method of improving automatic term extraction algorithms. *Procedia Computer Science*, 137:102–108.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tieyan Liu. 2020. Incorporating BERT into neural machine translation. In *International Conference on Learning Representations*.

A Appendix

For further comparability, we provide results of 15 prior term extraction methods provided by the ATR4S toolkit (Astrakhantsev, 2018). All methods provided by ATR4S are re-ranking methods based on a previous term candidate extraction process.

Table A1 shows the results of ATR4S on the ACTER heart-failure domain in English. While some methods achieve good precision, most methods show precision scores below our best models, even at only 100 terms extracted. Increasing the manually specified amount of k terms to extract results in a decrease of precision in favor of recall. The scores of the different methods level out towards the maximum of 2,000 terms extracted. The best F1 score is achieved by the DomainPertinence method at 2,000 terms extracted with an F1 score of 30.32%.

Table A2 shows the results of ATR4S on our ACLR2 test splits. One major drawback of prior methods is the required corpus size. The small test set in ACLR2 does not provide enough data for many of the statistical approaches or in fact the re-ranking to be effective at all after a certain amount of terms extracted. For the smaller Annotator 1 test set, we can observe virtually identical scores between all methods from 300 extracted terms onwards. For Annotator 2, this phenomena can be observed at 400 extracted terms. Best overall results are an F1 score of 21.83% for Weirdness at 200 terms extracted on the Annotator 1 test set and and F1 Score of 18.28% for Weirdness, PU and DomainPertinence at 300 terms extracted on the Annotator 2 test set. In comparison, our best models achieve an F1 score of over 75% for both Annotators.

ACTER: heart-failure EN (ATR4S)

Method	Top 100			Top 500			Top 1000			Top 2000		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
AvgTermFrequency	55.0	2.13	4.1	41.4	8.01	13.42	35.2	13.62	19.64	30.8	23.83	26.87
Basic	48.0	1.86	3.58	36.0	6.96	11.67	32.9	12.73	18.35	29.7	22.98	25.91
ComboBasic	48.0	1.86	3.58	36.0	6.96	11.67	31.9	12.34	17.8	29.7	22.98	25.91
CValue	54.0	2.09	4.02	39.6	7.66	12.84	33.7	13.04	18.8	32.1	24.84	28.0
DomainPertinence	58.0	2.24	4.32	46.8	9.05	15.17	35.8	13.85	19.97	**34.75**	**26.89**	**30.32**
KeyConceptRelatedness	**81.0**	**3.13**	**6.03**	**59.4**	**11.49**	**19.25**	**42.5**	**16.44**	**23.71**	31.1	24.06	27.13
LinkProbability	**83.0**	**3.21**	**6.18**	**73.0**	**14.12**	**23.66**	**52.8**	**20.43**	**29.46**	31.85	24.64	27.79
NovelTopicModel	49.0	1.9	3.65	39.8	7.7	12.9	34.9	13.5	19.47	29.95	23.17	26.13
PostRankDC	31.0	1.2	2.31	37.6	7.27	12.19	35.0	13.54	19.53	30.3	23.44	26.43
PU	61.0	2.36	4.54	46.2	8.94	14.98	38.3	14.82	21.37	**34.65**	**26.81**	**30.23**
Relevance	52.0	2.01	3.87	47.0	9.09	15.24	37.6	14.55	20.98	34.45	26.65	30.05
ResidualIDF	54.0	2.09	4.02	41.4	8.01	13.42	32.6	12.61	18.19	31.0	23.98	27.04
TotalTFIDF	31.0	1.2	2.31	39.8	7.7	12.9	36.7	14.2	20.47	31.05	24.02	27.09
Voting	**62.0**	**2.4**	**4.62**	**57.0**	**11.03**	**18.48**	**48.7**	**18.84**	**27.17**	**34.5**	**26.69**	**30.1**
Weirdness	31.0	1.2	2.31	38.0	7.35	12.32	37.4	14.47	20.86	31.65	24.49	27.61

Table A1: Precision, recall and F1 @ top *k* terms extracted of prior methods (ATR4S) on the English heart-failure domain texts of ACTER. Best three results per top *k* marked in bold.

ACL RD-TEC 2.0: Annotator 1 (ATR4S)

Method	Top 100			Top 200			Top 300			Top 400		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
AvgTermFrequency	36.0	7.53	12.46	29.0	12.13	17.11	28.73	16.53	20.98	28.73	16.53	20.98
Basic	40.0	8.37	13.84	36.0	15.06	21.24	28.73	16.53	20.98	28.73	16.53	20.98
ComboBasic	40.0	8.37	13.84	36.0	15.06	21.24	28.73	16.53	20.98	28.73	16.53	20.98
CValue	**47.0**	**9.83**	**16.26**	34.5	14.44	20.35	28.73	16.53	20.98	28.73	16.53	20.98
DomainPertinence	46.0	9.62	15.92	35.5	14.85	20.94	28.73	16.53	20.98	28.73	16.53	20.98
KeyConceptRelatedness	38.0	7.95	13.15	28.0	11.72	16.52	28.73	16.53	20.98	28.73	16.53	20.98
LinkProbability	40.0	8.37	13.84	29.0	12.13	17.11	28.73	16.53	20.98	28.73	16.53	20.98
NovelTopicModel	42.0	8.79	14.53	34.0	14.23	20.06	28.73	16.53	20.98	28.73	16.53	20.98
PostRankDC	43.0	9.0	14.88	36.0	15.06	21.24	28.73	16.53	20.98	28.73	16.53	20.98
PU	43.0	9.0	14.88	33.5	14.02	19.76	28.73	16.53	20.98	28.73	16.53	20.98
Relevance	46.0	9.62	15.92	35.5	14.85	20.94	28.73	16.53	20.98	28.73	16.53	20.98
ResidualIDF	36.0	7.53	12.46	29.0	12.13	17.11	28.73	16.53	20.98	28.73	16.53	20.98
TotalTFIDF	23.0	4.81	7.96	23.5	9.83	13.86	28.73	16.53	20.98	28.73	16.53	20.98
Voting	**47.0**	**9.83**	**16.26**	32.0	13.39	18.88	28.73	16.53	20.98	28.73	16.53	20.98
Weirdness	38.0	7.95	13.15	**37.0**	**15.48**	**21.83**	28.73	16.53	20.98	28.73	16.53	20.98

ACL RD-TEC 2.0: Annotator 2 (ATR4S)

Method	Top 100			Top 200			Top 300			Top 400		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
AvgTermFrequency	36.0	5.34	9.3	28.0	8.31	12.81	26.33	11.72	16.22	27.44	13.35	17.96
Basic	40.0	5.93	10.34	34.5	10.24	15.79	27.67	12.31	17.04	27.44	13.35	17.96
ComboBasic	40.0	5.93	10.34	34.5	10.24	15.79	27.67	12.31	17.04	27.44	13.35	17.96
CValue	44.0	6.53	11.37	34.0	10.09	15.56	29.0	12.91	17.86	27.44	13.35	17.96
DomainPertinence	38.0	5.64	9.82	**37.0**	**10.98**	**16.93**	**29.67**	**13.2**	**18.28**	27.44	13.35	17.96
KeyConceptRelatedness	35.0	5.19	9.04	31.0	9.2	14.19	27.33	12.17	16.84	27.44	13.35	17.96
LinkProbability	39.0	5.79	10.08	30.0	8.9	13.73	27.67	12.31	17.04	27.44	13.35	17.96
NovelTopicModel	39.0	5.79	10.08	34.5	10.24	15.79	27.67	12.31	17.04	27.44	13.35	17.96
PostRankDC	42.0	6.23	10.85	34.5	10.24	15.79	27.67	12.31	17.04	27.44	13.35	17.96
PU	42.0	6.23	10.85	34.0	10.09	15.56	28.0	12.46	17.25	27.44	13.35	17.96
Relevance	38.0	5.64	9.82	37.0	10.98	16.93	**29.67**	**13.2**	**18.28**	27.44	13.35	17.96
ResidualIDF	36.0	5.34	9.3	28.0	8.31	12.81	26.67	11.87	16.43	27.44	13.35	17.96
TotalTFIDF	19.0	2.82	4.91	24.0	7.12	10.98	26.33	11.72	16.22	27.44	13.35	17.96
Voting	**46.0**	**6.82**	**11.89**	36.0	10.68	16.48	29.0	12.91	17.86	27.44	13.35	17.96
Weirdness	38.0	5.64	9.82	29.5	8.75	13.5	**29.67**	**13.2**	**18.28**	27.44	13.35	17.96

Table A2: Precision, recall and F1 @ top *k* terms extracted of prior methods (ATR4S) on test split of ACLR2. Best result per top *k* marked in bold.