UB_ET at CheckThat! 2020: Exploring Ad hoc Retrieval Approaches in Verified Claims Retrieval

Edwin Thuma, Motlogelwa Nkwebi Peace, and Leburu-Dingalo Tebo and Mudongo Monkgogi

Department of Computer Science, University of Botswana
{thumae,motlogel,leburut,mudongom}@ub.ac.bw

Abstract. In this paper, we explore three different ad hoc retrieval approaches to rank verified claims, so that those that verify the input claim are ranked on top. In particular, we deploy DPH Divergence from Randomness (DFR) term weighting model to rank the verified claims. In addition, we deploy the Sequential Dependence (SD) variant of the Markov Random Fields (MRF) for term dependence to re-rank documents (verified claims) that have query terms (input claim) in close proximity. Moreover, we deploy LambdaMART, which is a learning to rank algorithm that use machine learning techniques to learn an appropriate combination of features into an effective ranking model.

Keywords: Check-Worthiness, Claim Retrieval, Proximity Search, Learning to Rank, Ad-hoc Retrieval

1 Introduction

Information posted on social media platforms such as Twitter is not often fact-checked by an authoritative entity before being published [2, 11]. In some instances, these posts on social media are coming from unreliable sources whose main objective is to disinform the general public. Such an action often yields undesirable results. For example, disinformation is often used in political campaigns in order to influence the outcome of political elections. It is for this reason that the Information Retrieval (IR) and the natural language processing community have invested significant effort in developing techniques to address disinformation, misinformation, factuality and credibility [2, 11]. This is evidenced by the CheckThat! lab[1], which is running under the Conference and Labs of the Evaluation Forum (CLEF)[2]. The CheckThat! Lab at CLEF 2020 is the third version of the lab. The other editions are the CheckThat! 2018 and CheckThat2019. The

[1] https://sites.google.com/view/clef2020-checkthat

[2] https://clef2020.clef-initiative.eu/

main purpose of these labs is to foster research in the development of techniques that would enable identification and verification of claims. In this paper, we present the results of our participation to the CheckThat! 2020 Task 2: Claim Retrieval, where we explore three different ad hoc retrieval approaches to rank verified claims, so that those that verify the input check-worthy tweet are ranked on top.

2 Background

In this section, we present a brief but essential background on the different ad-hoc retrieval approaches used in our investigation. In particular, we start by providing a description of the DPH term weighting model in Section 2.1. This is followed by a description of the learning to rank techniques in Section 2.2.

2.1 DPH Term Weighting Model

For all our experimental investigation, we used the parameter-free DPH term weighting model from the Divergence from Randomness (DFR) framework [1]. The DPH term weighting model calculates the score of a document d for a given query Q as follows:

$$score_{DPH}(d,Q) = \sum_{t\in Q} qtf \cdot norm \cdot \left(tf \cdot \log((tf \cdot \frac{avg\_l}{l}) \cdot (\frac{N}{tfc})) + 0.5 \cdot \log(2 \cdot \pi \cdot tf \cdot (1 - t_{MLE}))\right) \quad (1)$$

where qtf, tf and tfc are the frequencies of the term t in the query Q , in the document d and in the collection C respectively. N is number of documents in the collection C, avg_l is the average length of documents in the collection C and l is the length of the document d. $t_{MLE} = \frac{tf}{l}$ and $norm = \frac{(1-t_{MLE})^2}{tf+1}$.

2.2 Learning to Rank Approach

Learning to rank techniques are algorithms that use machine learning techniques to learn an appropriate combination of features into an effective ranking model [4]. This effective ranking model can be leant through the following steps [5, 6]:

1. Top K retrieval: Using a set of training queries that have relevance assessment, retrieve a sample of k documents using an initial weighting model such as DPH.
2. Feature extraction: For each document in the retrieved sample, extract a set of features. These features can either be query-dependent (term weighting models, term dependence models) or query-independent (click count, fraction of stopwords). The feature vector for each document is labelled according to the already existing relevance judgements.
3. Learning: Learn an effective ranking model by deploying an effective learning to rank technique on the feature vectors of the top k documents.

The learned model can be deployed in a retrieval setting as follows:

4. Top K retrieval: For each unseen query, the top k documents are retrieved using the same retrieval strategy as in step (1)
5. Feature extraction: A set of features are extracted for each document in the sample of k documents. These features should be the same as those extracted in step (2).
6. Re-rank the documents: Re-rank the documents for the query by applying a learned model on every feature vector of the documents in the sample. The final ranking of the documents is obtained by sorting the predicted scores in descending order.

In this work, we deploy LambdaMART [3], which is a tree-based learner. A tree-based learner builds a set of regression trees T. The final score of a document d is obtained by traversing the nodes of a particular tree t, according to the decisions based on the vector of feature values of the document f_d [3, 6]. The leaf node of the tree traversed represents the final score of the document d. This can be expressed as:

$$score(d, Q) = \sum_{t \in T} t(f_d) \tag{2}$$

3 Experimental Setting

FAQ Retrieval Platform: For all our experiments, we used Terrier-4.2[3] [8], an open source Information Retrieval (IR) platform. All the documents used in this study were first pre-processed before indexing and this involved tokenising the text and stemming each token using the full Porter stemming algorithm [10]. We indexed the collection using blocks in order to save positional information with each term.
Training Learning to Rank Techniques: For our learning to rank approach, we used the Terrier-4.2 Fat[4][6] framework. Fat is a method of allowing many features to be computed within one run of Terrier. To train and test LambdaMART, we used the default parameter values of the algorithms.

4 Description of the Different Runs

T2-EN-UB_ET-DPH: For all our runs, we used the parameter-free DPH Divergence From Randomness term weighting model in Terrier-4.2 IR platform to score and rank the documents (verified claims)

T2-EN-UB_ET-DPH_LTR: We used **T2-EN-UB_ET-DPH** as the baseline system. As improvement, we deployed a learning to rank technique. For our

[3] http://terrier.org/
[4] http://terrier.org/docs/v4.0/learning.html

learning to rank technique, we used the training and development tweets with their qrels for training and validation. We used the Terrier-4.2 Fat framework to retrieve 1000 documents for each topic (tweet) using the DPH term weighting model, and then calculated several additional query dependent features in Table 1. Using these features, we used Jforests to learn a LambdaMART model. We then applied this learned model on the test tweets to generate a final ranking.

Table 1. All query-dependent (QD) features used in this work.

Features	**Type**	**Total**
Weighting models (BM25, PL2 and TF-IDF)	QD	3
Proximity (Dependence) Models (DFRDependenceScoreModifier [9] and MRFDependenceScoreModifier [7])	QD	2
Total		5

T2-EN-UB_ET-DPH_MRF: We used **T2-EN-UB_ET-DPH** as the baseline system. As improvement, we deployed the Sequential Dependence (SD) variant of the markov random field for term dependence [7] to re-rank documents (verified claims) that have query terms (input claim) in close proximity. Sequential Dependence only assumes a dependence between neighbouring query terms.

5 Results and Discussion

Table 2. Task 2, English: Performance for all the 3 Runs

Run ID	MAP@1	MAP35	MAP@5	P@1	P@3	P@5
T2-EN-UB_ET-DPH	0.843	0.868	**0.873**	0.840	0.300	0.185
T2-EN-UB_ET-DPH_LTR	0.818	0.862	0.864	0.815	0.307	**0.186**
T2-EN-UB_ET-DPH_MRF	0.838	0.865	0.869	0.835	0.300	0.184

Table 2 presents our evaluation results. The official evaluation measure for Task 2: Claim Retrieval is MAP@k, where $k = 5$. We also present Precision@k. The results of this study suggests that ad-hoc retrieval approaches such as term weighting models, proximity (Dependence) models and learning to rank techniques can be used to rank verified claims for a given check-worthy tweet. Overall, our primary submission **T2-EN-UB_ET-DPH_LTR** ranked third out of 10 submissions. It is worth noting that an attempt to improve the retrieval performance using a learning to rank technique resulted in a degradation in performance. An examination of the data revealed that for a majority of check-worthy tweets, there was a single verified claim. This lack off sufficient training data could have resulted in the degradation in retrieval performance. For example, after performing a first-pass retrieval with DPH and attempting to improve the ranking with our learned ranking model, some verified claims that verify the input claim ranked lower than in the previous ranking.

6 Conclusion

In this paper, three different ad hoc retrieval approaches were explored to determine their effectiveness in raking verified claims so that those that verify the input claim are ranked on top. The results of this study suggests that term weighting models such as DPH can be used to rank verified claims for a given check-worthy tweet. In our attempt to improve the retrieval effectiveness using a learning to rank technique, we noticed a degradation in retrieval performance. In future, we will explore using sufficient training data in our learning to rank technique coupled with additional query dependent and query independent features. Similarly, re-ranking the verified claims using markov random field for term dependence resulted in the degradation in performance. In our experiments, default parameter settings were used. Further research could usefully explore using different parameters settings such as varying the window size in order to improve the retrieval performance.

References

1. G. Amati, E. Ambrosi, M. Bianchi, C. Gaibisso, and G. Gambosi. FUB, IASI-CNR and University of Tor Vergata at TREC 2007 Blog Track. In *Proceedings of the 16th Text REtrieval Conference (TREC-2007)*, pages 1–10, Gaithersburg, Md., USA., 2007. Text REtrieval Conference (TREC).
2. Alberto Barrón-Cedeño, Tamer Elsayed, Preslav Nakov, Giovanni Da San Martino, Maram Hasanain, Reem Suwaileh, Fatima Haouari, Nikolay Babulkov, Bayan Hamdan, Alex Nikolov, Shaden Shaar, and Zien Sheikh Ali. Overview of CheckThat! 2020: Automatic identification and verification of claims in social media.
3. Yasser Ganjisaffar, Rich Caruana, and Cristina Lopes. Bagging gradient-boosted trees for high precision, low variance ranking models. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information*, SIGIR '11, pages 85–94, New York, NY, USA, 2011. ACM.
4. T.-Y. Liu. Learning to Rank for Information Retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, June 2009.
5. C. Macdonald, R.L. Santos, and I. Ounis. The whens and hows of learning to rank for web search. *Information Retrieval*, 16(5):584–628, October 2013.
6. C. Macdonald, R.L.T. Santos, I. Ounis, and B. He. About learning models with multiple query-dependent features. *ACM Transactions on Information Systems (TOIS)*, 31(3):11:1–11:39, August 2013.
7. Donald Metzler and W. Bruce Croft. A markov random field model for term dependencies. In Ricardo A. Baeza-Yates, Nivio Ziviani, Gary Marchionini, Alistair Moffat, and John Tait, editors, *SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, August 15-19, 2005*, pages 472–479. ACM, 2005.

8. I. Ounis, G. Amati, Plachouras V., B. He, C. Macdonald, and Johnson. Terrier Information Retrieval Platform. In *Proceedings of the 27th European Conference on IR Research*, volume 3408 of *Lecture Notes in Computer Science*, pages 517–519, Berlin, Heidelberg, 2005. Springer-Verlag.
9. V. Plachouras and I. Ounis. Multinomial Randomness Models for Retrieval with Document Fields. In *Proceedings of the 29th European Conference on IR Research*, pages 28–39, Berlin, Heidelberg, 2007. Springer-Verlag.
10. M.F. Porter. An Algorithm for Suffix Stripping. *Readings in Information Retrieval*, 14(3):313–316, 1997.
11. Shaden Shaar, Alex Nikolov, Nikolay Babulkov, Firoj Alam, Alberto Barrón-Cedeño, Tamer Elsayed, Maram Hasanain, Reem Suwaileh, Fatima Haouari, Giovanni Da San Martino, and Preslav Nakov. Overview of CheckThat! 2020 English: Automatic identification and verification of claims in social media.