Detecting crisis-related tweets by example

Anna Kruspe*
German Aerospace Center (DLR)
anna.kruspe@dlr.de

Jens Kersten
German Aerospace Center (DLR)

Friederike Klan
German Aerospace Center (DLR)

v1.1.1
2018/09/10

ABSTRACT

Messages obtained from social media sources can be extremely helpful in crisis situations, but discovering relevant ones in a flood of other data is not trivial. Methods have so far focused on universal detection models for crises or specific types of them. Event-specific models would fare better than those, but data collection and training is costly and may take too much time for realtime analysis and response.
As a compromise, collecting a small amount of example messages manually is feasible and can be performed through a search for certain keywords or hashtags. Few-shot models can generalize to new examples with such a small handful of examples, and do not need be trained anew for each event. We show how such models can be used to detect crisis-relevant tweets with just 10 to 100 examples and counterexamples. We also apply a new type of few-shot model that does not require counterexamples (one-way prototypical networks).

Keywords

Social media, Twitter, Relevance, Keywords, Hashtags, Few-shot models, One-class models

MOTIVATION

Social media is an interesting source of information during disasters that has so far been largely untapped. Twitter users, as an example, write about disaster preparations, developments, recovery, and a host of other topics (Niles et al. 2018). Retrieving this information could lead to significant improvements in disaster management strategies. According to a Red Cross study, 69% of Americans think that emergency response agencies should respond to calls for help sent through social media channels (American Red Cross 2010).
The crux of this matter lies in the retrieval and classification of such messages. Twitter users generate 5,700 tweets per second on average[1]. Even with hashtag- or location-based pre-filtering, sophisticated automatic methods are necessary to detect disaster-related messages in acceptable timespans.
Approaches published so far have mainly had the generalized detection of any disaster-related tweet as their objective (Burel and Alani 2018), or have focused on specific types of disasters (Caragea et al. 2016). Designing models specific to a certain event could potentially yield much more exact results, but training such dedicated models would require large amounts of already available data and take up critical time in a disaster.
In this paper, we present a first approach that allows for the detection of Twitter messages (tweets) pertaining to a specific event on the basis of some example tweets. These models can be trained well in advance, and then require as few as ten examples against which new tweets are compared. The paper is organized as follows: In the next two sections, we will present an overview of related work and the data sets used in our experiments. The sections afterwards describe the proposed approaches and our experimental results. In the last section, we give a conclusion of our work so far and make suggestions for future experiments.

*corresponding author
[1] https://blog.twitter.com/engineering/en_us/a/2013/new-tweets-per-second-record-and-how.html

RELATED WORK

As described above, users generate huge amounts of data on Twitter every second, and finding tweets related to an ongoing event is not trivial (Landwehr and Carley 2014). Several detection approaches have been presented in literature so far.

The most obvious strategy is the filtering of tweets by various surface characteristics as shown in (Kumar et al. 2011), for example. Keywords and hashtags are used most frequently for this and often serve as a useful pre-filter. Olteanu et al. developed a lexicon called *CrisisLex* for this purpose (Olteanu, Castillo, et al. 2014). However, this approach easily misses tweets that do not mention the keywords specified in advance, particularly when changes occur or the attention focus shifts during the event. It may also retrieve unrelated data that contains the same keywords (Imran, Castillo, et al. 2015). Geo-location is another frequently employed feature that can be useful for retrieving tweets from an area affected by a disaster. However, this approach misses important information that could be coming from a source outside the area, such as help providers or news sources. Additionally, only a small fraction of tweets is geo-tagged at all, leading to a large amount of missed tweets from the area (Sloan et al. 2013).

To resolve these problems, several other strategies were developed, starting with crowdsourcing platforms. On these platforms, a large amount of users hand-selects and labels incoming tweets in disaster situations. Examples include *Ushahidi*[2] and *CrisisTracker* (Rogstadius et al. 2013). Some of them already integrate "traditional" machine learning models, e.g. *AIDR* (Imran, Castillo, et al. 2015).

In recent years, approaches based on deep learning techniques have come to the forefront of research. On a more general level, the problem falls under the umbrella of event detection as shown, for example, in (Chen et al. 2015; Feng et al. 2016; T. H. Nguyen and Grishman 2015). Caragea et al. first employed Convolutional Neural Networks (CNN) for the classification of tweets into those related to flood events and those unrelated (Caragea et al. 2016). In many of the following approaches, a type of CNN developed by Kim for text classification is used (Kim 2014), such as in (Burel and Alani 2018). This method achieves an accuracy of 80% for the classification into related and unrelated tweets. In this publication as well as in (Burel, Saif, et al. 2017) and (D. T. Nguyen et al. 2016), the same kind of model is also used for information type classification.

What all of these approaches have in common is that they aim to find tweets that are related or relevant or possibly informative with regards to crisis events. These qualities are not easy to define, and might vary for different users or different types of events. Moreover, enabling a model to detect related tweets without any a-priori information about an event is challenging. A real-world system may not need to be restricted in this way; in many cases, its users will already have some information about the event, and may already have spotted tweets of the required type. This removes the need to anticipate any type of event. On the other hand, it directs the system towards a specific event rather than any event happening at that time. In this paper, we demonstrate how to implement such a system.

DATA

We employ two widely-used collections of disaster-related tweets to train and test our models: CrisisLexT26 and *CrisisNLP*.

CrisisLexT26

CrisisLexT26 was first published by Olteanu et al. in 2014 (Olteanu, Castillo, et al. 2014) and expanded later (Olteanu, Vieweg, et al. 2015). It contains tweets collected during 26 crises, mainly natural disasters like earthquakes, wildfires and floods, but also human-induced disasters like shootings and a train crash. Amounts of these tweets per disaster range between 1,100 and 157,500. In total, around 285,000 tweets were collected. They were then annotated by paid workers on the *CrowdFlower* crowdsourcing platform[3] according to three concepts: Informativeness, information type, and tweet source. In this work, a balanced set containing 1,100 English-language tweets per event is used.

CrisisNLP

Similar to *CrisisLexT26*, the team behind *CrisisNLP* collected tweets during 19 natural and health-related disasters and published them for research (Imran, Mitra, et al. 2016). Collected tweets range between 17,000 and 28 million per event, making up around 53 million in total. Out of these, around 50,000 were annotated both by volunteers and by paid workers on *CrowdFlower* with regard to information type. In this work, only the tweets with *CrowdFlower* annotations were utilized. These tweets come from a subset of 11 English-language event sets, summing up to around 23,000 in total. There is no overlap between the events in both data sets.

[2] https://www.ushahidi.com/

[3] Now named *Figure Eight*, https://www.figure-eight.com/

Event	Hashtag	Occurrences/1001
2012 Typhoon Pablo	#PabloPH	453
2013 Bohol Earthquake	#PrayForVisayas	338
2013 Singapore Haze	#sghaze	667
2013 West Texas Explosion	#PrayForTexas	152
2012 Italy Earthquakes	#terremoto	711
2013 Manila Floods	#MaringPH	399
2013 Boston Bombings	#PrayForBoston	259
2013 Brazil Nightclub Fire	#SantaMaria	353
2013 Colorado Floods	#coflood	317
2013 LA Airport Shootings	#LAX	451
2012 Guatemala Earthquake	#sismo	165
2012 Philippines Floods	#rescuePH	571
2013 Sardinia Floods	#Sardegna	764
2012 Venezuela Refinery Explosion	#Amuay	588
2013 Alberta Floods	#yycflood	497
2013 Lac Megantic Train Crash	#LacMegantic	254
2013 Typhoon Yolanda	#Haiyan	264
2013 Glasgow Helicopter Crash	#Clutha	286
2013 Queensland Floods	#bigwet	684
2012 Colorado Wildfires	#colorado	151
2013 Australia Bushfire	#nswfires	481
2013 Savar Building Collapse	#Bangladesh	579
2012 Costa Rica Earthquake	#earthquake	363
2013 Russia Meteor	#RussianMeteor	407

Table 1. Hashtags chosen for the *CrisisLexT26* data set.

Experimental data composition

In order to train and test our few-shot models, sub-sets of positive and negative supports for a class (examples and counterexamples) plus an either positive or negative query are necessary. In few-shot models, training steps performed on such sets are called "episodes". Tweets are considered to belong to the same class if they are associated with the same event. Tweets with annotations marking them as unrelated or irrelevant are filtered. Otherwise, the annotations are not used. Event selection for each episode is performed in such a way that all events are equally likely.

To generate episode data packs, tweets for an event containing a specific hashtag are selected randomly to create the positive support set. Case is ignored, and this hashtag is removed in a pre-processing step so as not to bias the classifier towards it. For positive queries, tweets without the hashtag, but coming from the same event are picked randomly. For negative supports and queries, random tweets from other events are used. Hashtags were chosen manually according to their prevalence in the event's tweets. An overview is given in tables 1 and 2. The practical reasoning for this is that during an ongoing event, a user could quickly search for example tweets by a certain hashtag, then use the found examples to detect more event-related tweets using the models proposed by us. In the few-shot experiments, *CrisisLexT26* is used to generate the training episodes, and *CrisisNLP* is used for the validation episodes. In this way, we ensure that our results represent the generalization ability of the models rather than their ability to adapt to a set of specific events.

PROPOSED APPROACHES

We propose three approaches for detecting event-related tweets: Models trained explicitly on data related to a specific event, matching networks trained on event-related data and counterexamples, and one-way prototypical networks only trained on event-related data.

Dedicated event models

As an anchor experiment, we trained event-wise models directly on all tweets containing the event-related hashtag plus random tweets from other events. We then tested them on tweets from the same event without the hashtag, and random unrelated tweets from other events.

The architecture used in this experiment is a Convolutional Neural Network (CNN) as proposed by Kim (Kim

Event	Hashtag	Occurrences/Total
2013 Pakistan Earthquake	#Balochistan	402/2015
2014 California Earthquake	#napa	272/2016
2014 Chile Earthquake	#PrayForChile	452/2014
2014 Ebola Virus	#ebola	999/2018
2014 Hurricane Odile	#Odile	440/2016
2014 India Floods	#india	250/2009
2014 MERS	#MERS	589/2021
2014 Pakistan Floods	#KashmirFloods	218/2016
2014 Typhoon Hagupit	#hagupit	439/2016
2015 Cyclone Pam	#vanuatu	336/2014
2015 Nepal Earthquake	#NepalQuake	309/3022

Table 2. Hashtags chosen for the *CrisisNLP* data set.

2014). A visualization is provided in figure 1. This approach was originally developed for classifying sentences into semantic categories, e.g. question types or sentiments. In a pre-processing step, each tweet is padded with empty units or shortened to contain exactly 25 words. At the input, these words are transformed into word embeddings using pre-trained weights, which we keep static througout training. We use the pre-trained, crisis-specific *word2vec* embeddings described in (D. T. Nguyen et al. 2016). Then, several convolutional layers with different kernel widths are applied in parallel. Global max-pooling is performed for each of these layers, and the results are concatenated. This new embedding is then fed into a 128-dimensional fully connected layer with dropout and a subsequent softmax layer to determine the final class. This type of model has successfully been used for crisis-related data (Burel and Alani 2018; D. T. Nguyen et al. 2016).
We expect these models to perform very well as they are dedicatedly adapted to a certain event. However, this is almost never an option in a real-world situation because it would require the collection of a large amount of related data in advance, which is not possible until social media users have been discussing the event to some extent. Additionally, a new model has to be trained from scratch, which uses up even more valuable time during an event. The resulting model is also not very flexible; if discussion topics within the event shift, it may not perform well anymore. For these reasons, this experiment serves merely as an upper boundary for the accuracy of a model trained on event-specific hashtags.

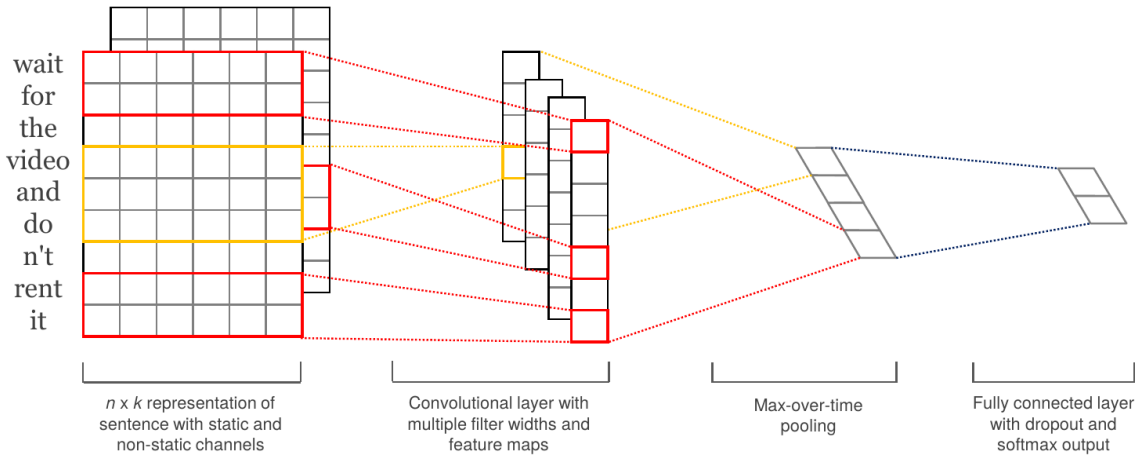

Figure 1. CNN for text classification as proposed by Kim (Kim 2014).

Matching networks

Matching networks which implement one-shot learning were first introduced by Vinyals et al. in 2016 (Vinyals et al. 2016). The concept of one- or few-shot learning is motivated by human learning: If humans are presented with a new class of objects, they do not require more than a few examples to be able to match new instances to this class (often, just one supporting example is sufficient). This ability is, among other factors, informed by general world

experience, which teaches humans to set boundaries between object classes.
Matching networks are thus trained on sets of support examples for a subset of possible classes plus a query example belonging to one of the classes in so-called episodes. These episodes are generated for a wide range of class permutations. In the one-shot case, only one support example is given for each class, while there are multiple ones in few-shot models.
An example of the basic structure of one-shot matching networks is shown in figure 2. The network has two input branches: One for the support examples, and one for the query example. In both branches, the inputs are run through a sub-network which performs an embedding (expressed as embedding functions g_θ and f_θ). The embedding networks may or may not share their weights. The query's embedding is then compared against the supports' embeddings using a pre-defined distance metric. This results in likelihood measures for each input class, commonly implemented with a softmax. The over-all comparison metric for the task is learned implicitly as the network learns an appropriate embedding.
In the few-shot case, multiple support examples per class are treated independently of each other throughout the network. Example-related likelihoods are then summed into class likelihoods at the output. In this way, matching networks essentially implement a weighted nearest-neigbor classification.
Few-shot learning has been used for short-text classification in (Yan et al. 2018). The architecture is slightly different, employing a CNN-based Siamese network with hinge loss as the distance metric and a subsequent SVM classifier. Support examples for one class are combined into a prototype (also see next section). The approach is tested for sentiment classification of tweets, among other tasks, where it outperforms other deep learning models.
For our experiments, we employ a relatively straightforward matching network architecture, which is illustrated in figure 3. The network has one input branch for a number of support examples for each class, and another for one query example. The subsequent CNN is identical to the one described in the previous section, except that there is only one 512-dimensional fully connected layer without a softmax function after the max-pooling layers. Its outputs are used as the internal embeddings in the further processing. They are fed into a distance metric layer, where the query embedding is compared to each support embedding. As proposed in (Vinyals et al. 2016), we use the cosine distance as our metric here. The resulting distances are then processed by a softmax layer and summed up for each class, generating the final class scores.
Because we are applying matching networks to a binary problem (a query tweet is either related the considered event or not), we need support examples for two classes: Positive examples of tweets concerning the event, and random other tweets. Semantically, this amounts to a workaround, as there are no two comparable classes. Instead, one of them merely consists of counterexamples (a sort of "garbage" class). Examples in this class will have a much wider feature distribution. This makes selecting random examples difficult, as they should still be representative of this distribution, particularly when only a small number of support examples is used.

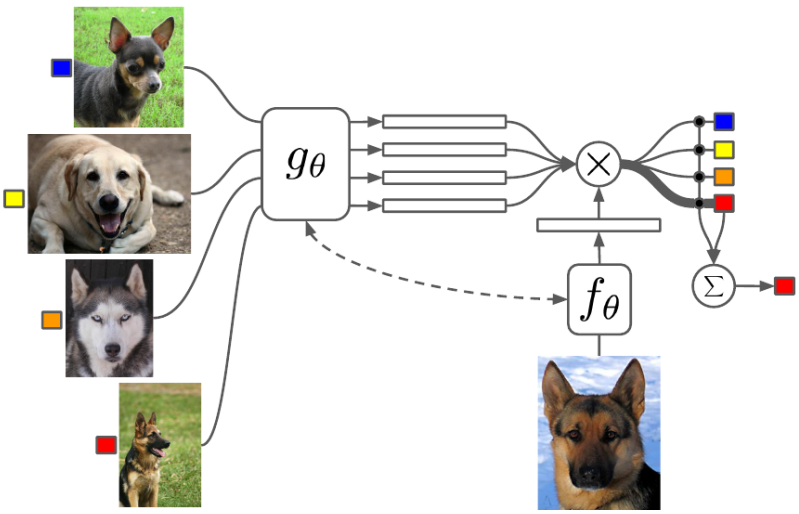

Figure 2. Matching network architecture as proposed by Vinyals et al. (Vinyals et al. 2016).

One-way prototypical networks

Prototypical networks are an extension to matching networks and were proposed by Snell et al. in 2017 (Snell et al. 2017). Instead of treating all of a class' support examples independently in the few-shot case, they compute a so-called "prototype" of the class after the embedding network. This prototype is commonly the mean of the class;

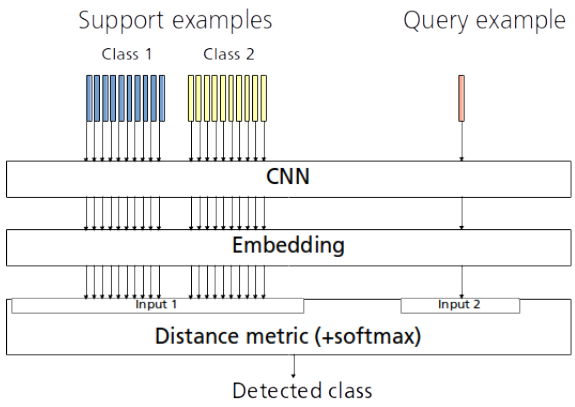

Figure 3. Matching network architecture used in our experiments.

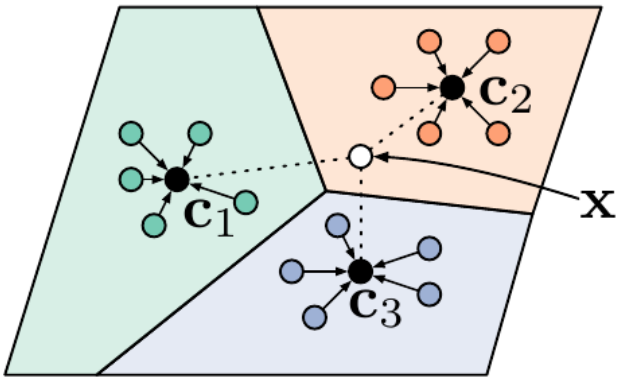

Figure 4. Calculation of class prototypes c_k by averaging support embeddings as proposed by Snell et al. (Snell et al. 2017).

a visualization is provided in figure 4. When the squared Euclidean distance is used as the metric, this implements a linear classifier (as opposed to the nearest neighbor classifier produced by matching networks).
We take this approach one step further. As described above, using counterexamples as one class in a binary few-shot problem presents some difficulties. We therefore construct a prototypical network that only requires positive support examples, from which it generates an internal embedding of the whole class. The query embedding is then compared to this, and judged to belong to the same class or not. We call this type of network a "one-way prototypical network". The sub-network for generating the internal embeddings is the same as before. An illustration is shown in figure 5. This approach presents a challenge for the distance metric: Because Cosine or Euclidean distances are not bounded, we cannot set a threshold for class belonging. We solve this by substituting a novel Gaussian probability layer. In this layer, Gaussian distributions are fitted over the support embeddings for each embedding dimension. Then, the probability of each of the query embedding's dimensions is calculated and normalized. The average of these probabilities is used as the final result.

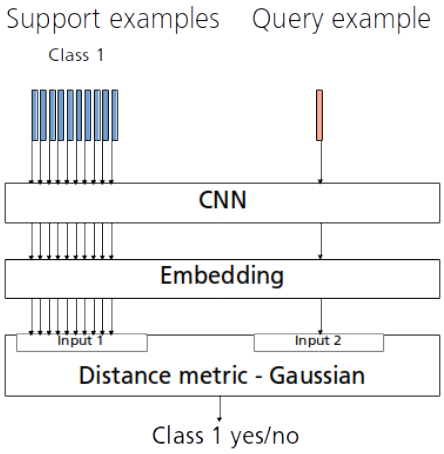

Figure 5. One-way prototypical network architecture used in our experiments.

EXPERIMENTAL RESULTS

In this section, we present the experimental set-ups and results for each of the three proposed approaches.

Dedicated event models

As described in the previous section, we start by training dedicated models for each event in the *CrisisLexT26* data set. Training is performed on tweets containing the matching hashtag from table 1 and unrelated examples from other events. The models are then tested on tweets from the considered event without the hasthtag.
The results are shown in figure 6. As expected, training individual models on the data specific to an event works very well due to the focus on a small set of topics or keywords. Accuracies range between 83% and 100%, with an average of 94%. Events of types that occur multiple times in the data set produce somewhat lower values (e.g. floods, earthquakes), suggesting at least a partial focus of the model on the event type rather than the specific event. This is to be expected and not necessarily a problem for practical purposes, unless two events of the same type occur at the same time. Other influences may include the use of other languages than English in the event data, and the range of topics of interest during an event, which is also related to the duration of the event.
Despite the high accuracies achieved with dedicated models, this approach will often be infeasible during an actual event. A significant amount of data for the event needs to be collected first, and then model training has to be performed and optimized. This strategy could be implemented for events that unfold over a longer timespan; however, this introduces the problem of shifting topics during the course of the event, for which the model may not be well-adapted anymore.

Matching networks

In our second set of experiments, we train matching networks. Training episodes are generated from random tweets from *CrisisLexT26* containing the matching hashtag from table 1 as the positive supporting examples, and random tweets from other events as the negative ones; queries are generated from event-related tweets without the hashtag, and once again random tweets from other events. This model is then evaluated on the *CrisisNLP* data set, where test

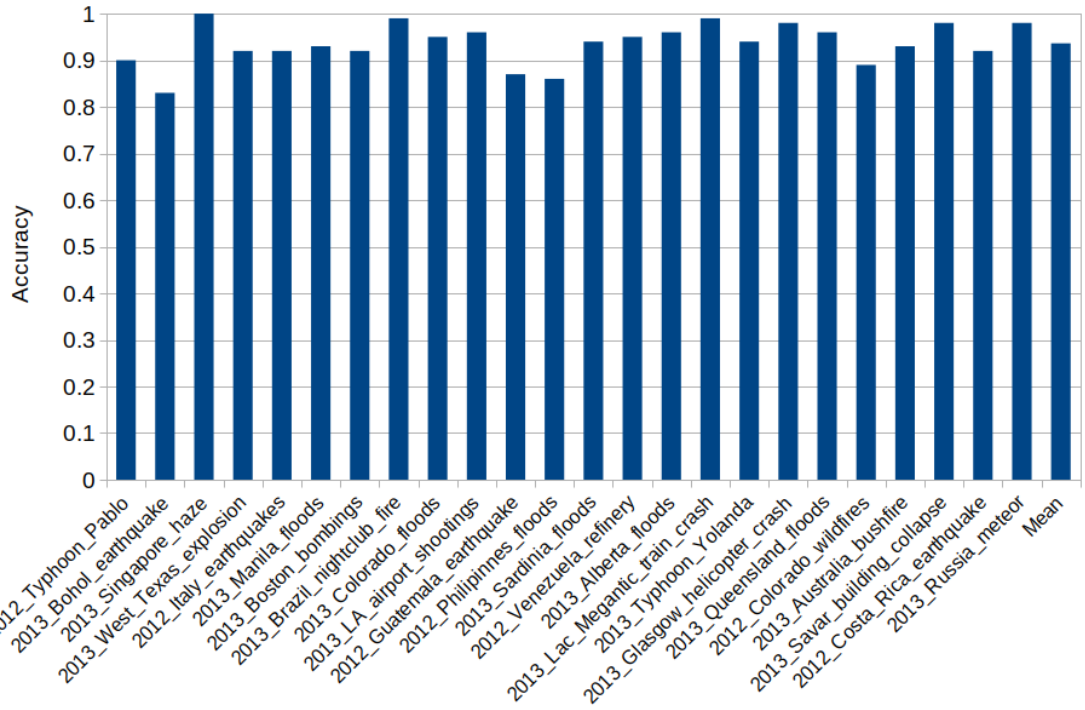

Figure 6. Results of the dedicated event model experiments. Accuracy is shown for each individual event; the last bar is the mean.

episodes are generated in the same way.
The results are shown in figure 7. In total, 40 configurations are tested: Training is performed with 1280 to 12800 episodes (colored bars), and the number of support examples per class is varied between 10 and 100. As a general tendency, the accuracy improves with more support examples and with more training episodes. However, as few as 20 support examples already produce an accuracy of 72% when trained on a sufficient number of episodes. The results become slightly higher up to 80 or 90 support examples. Providing more support examples improves the likelihood that a query tweet will be similar to one of the support examples. In our use case, the main difficulty for this lies in the negative queries, as those will be very diverse, just as the negative support examples.
Considering the amount of training episodes, accuracy usually does not improve beyond 6400. At this point, many of the possible permutations may already be covered in the randomly generated episodes. Further episodes then do not provide more information and just serve to effectively increase the number of training epochs. Of course, this is dependent on the size of the data set.
Over-all, the results do not vary highly across parameter configurations, and their tendencies are somewhat noisy. We believe that this has to do with the random creation of the episodes, especially with regards to negative examples. However, at a top accuracy of 73% on completely unseen events, this approach is feasible for use in a real-world system for the detection of event-related tweets. With no necessary re-training, the model can be used out-of-the-box for new events. The result is somewhat lower than that of the state of the art for detecting related tweets (Burel and Alani 2018); however, our approach is specific to a certain event versus other events.

One-way prototypical networks

Finally, we perform experiments with one-way prototypical networks on the same data. As described above, these networks only require positive examples for the considered event, removing the issue of choosing a representative set of counterexamples. Once again, models are trained on 1280 to 12800 automatically generated episodes with 10 to 100 support examples for the one class. Figure 8 shows the results.
The best achieved accuracy is 72%, demonstrating that this new type of network works just as well for our purposes as the previously existing matching networks. The trends are the same as in the previous experiment: More training episodes generally improve accuracy, although further experiments show that this effect diminishes beyond 12800. More support examples also help, but this does not change much between 40 and 100. Since this type of model calculates a Gaussian distribution on the support examples' internal embeddings, we assume that these distributions can be estimated well enough from 40 examples or even fewer. One effect that stands out is that these trends are

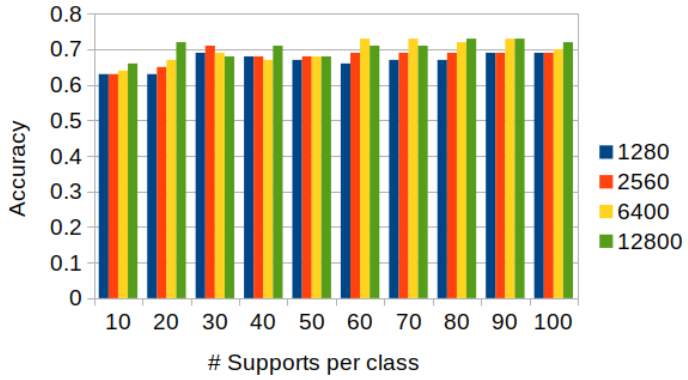

Figure 7. Results of the matching network experiments in terms of accuracy vs. number of support examples per class; colored bars represent the number of training episodes.

much clearer here. This is probably the case because of the non-reliance on counterexamples. These introduce noise to the process, which might make the model work better or worse in some cases because of the selection of these examples and their distribution and relation to the positive examples.
Just like the previously analyzed matching network models, this method is well-suited for practical operation as no re-training is necessary for a new event. In contrast to the previous approach, only half as many support examples need to be collected, and no attention has to be paid to finding widely distributed counterexamples.
In an additional experiment (not shown here), we also tried using tweets not related to any disaster as the negative queries. This improved the accuracy to around 80%, which is around the state of the art for the relatedness task (Burel and Alani 2018). In many practical cases, this model will be sufficient, as users will often not need to distinguish events from each other if there is only one happening at that time. Despite some thematic overlap, this model will still be more focused on the event for which support examples are given rather than all events.

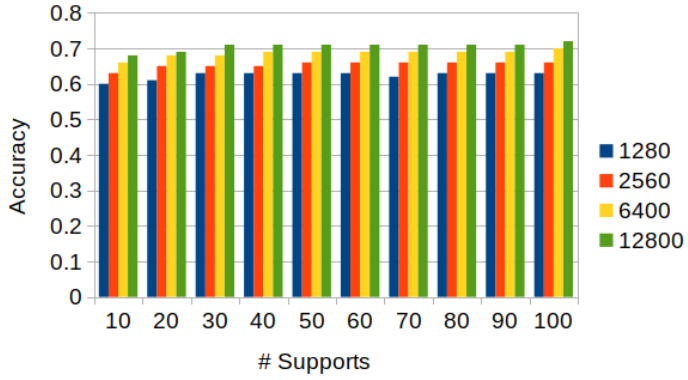

Figure 8. Results of the one-way prototypical network experiments in terms of accuracy vs. number of support examples per class; colored bars represent the number of training episodes.

CONCLUSION & FUTURE WORK

In this paper, we demonstrate three approaches for detecting tweets related to a specific crisis. In the first one, we train dedicated models for this crisis. This results in a mean accuracy of 93%. For the second experiment, we train matching networks for sets of 10 to 100 support examples and counterexamples for the considered event. We then develop this approach further by training one-way prototypical networks, which do not require counterexamples anymore. The second and third method produce accuracies of 73% and 72% respectively.
All three approaches are good enough to potentially be used as a filtering mechanism in crisis situations. However, training dedicated crisis models will often be infeasible as this takes up too much time and effort for a quick response. Both of the other approaches can be used out-of-the-box in a new scenario, only requiring some example tweets. As the results produced by the matching network show, the selection of the counterexamples can have an effect on the over-all effectiveness of the model, depending on their variation and relatedness to the positive examples. The one-way prototypical network removes this issue and is therefore even easier to integrate into a live system.

Our experiments on the matching networks and one-way prototypical networks show that the number of support examples and the number of training episodes have a small effect on the test accuracy. In the case of the support examples, this effect disappears beyond 40, probably because this amount of support examples already covers the semantic space well in many cases. For the number of training episodes, we also observed diminishing returns after a certain number because many permutations will already have occurred at that point.
Some easy adaptations could be made to the processing stages of these models for a real-world system. Depending on whether better precision or better recall is required, the decision boundary can be adjusted accordingly. We also performed additional experiments (not shown here) on the same systems trained on support data for various event versus non-event data. This improved the accuracy to around 80%, and may often be more suitable for a practical system.
In the future, these methods could be improved in a number of ways. Ideally, they could be trained on a much wider range of (disaster) events as this would help their generalization abilities towards unseen events. When possible, the counterexamples could be picked in a more directed fashion to stabilize the networks' decision boundaries, or the network could simply be enabled to process a much larger amount of counterexamples than positive examples without changing the class priors. Finally, the one-way prototypical network method itself could be expanded in multiple ways, e.g. by replacing the Gaussian probability layer with a Gaussian Mixture Model.

REFERENCES

American Red Cross (Aug. 2010). *Social Media in Disasters and Emergencies*.

Burel, G. and Alani, H. (May 2018). "Crisis Event Extraction Service (CREES) - Automatic Detection and Classification of Crisis-related Content on Social Media". In: *15th International Conference on Information Systems for Crisis Response and Management*. Rochester, NY, USA.

Burel, G., Saif, H., and Alani, H. (Oct. 2017). "Semantic Wide and Deep Learning for Detecting Crisis-Information Categories on Social Media". In: *International Semantic Web Conference (ISWC)*. Vienna, Austria.

Caragea, C., Silvescu, A., and Tapia, A. H. (May 2016). "Identifying informative messages in disaster events using Convolutional Neural Networks". In: *13th International Conference on Information Systems for Crisis Response and Management (ISCRAM)*. Rio de Janeiro, Brazil.

Chen, Y., Xu, L., Liu, K., Zeng, D., and Zhao, J. (July 2015). "Event Extraction via Dynamic Multi-Pooling Convolutional Neural Networks". In: *53rd Annual Meeting of the Association for Computational Linguistics (ACL)*. Beijing, China.

Feng, X., Qin, B., and Liu, T. (Aug. 2016). "A language-independent neural network for event detection". In: *54th Annual Meeting of the Association for Computational Linguistics (ACL)*. Berlin, Germany.

Imran, M., Mitra, P., and Castillo, C. (May 2016). "Twitter as a lifeline: Human-annotated twitter corpora for NLP of crisis-related messages". In: *Tenth International Conference on Language Resources and Evaluation (LREC)*. Portoroz, Slovenia.

Imran, M., Castillo, C., Diaz, F., and Vieweg, S. (June 2015). "Processing Social Media Messages in Mass Emergency: A Survey". en. In: *ACM Computing Surveys* 47.4, pp. 1–38.

Kim, Y. (Oct. 2014). "Convolutional neural networks for sentence classification". In: *2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar.

Kumar, S., Barbier, G., Abbasi, M. A., and Liu, H. (2011). "TweetTracker: An Analysis Tool for Humanitarian and Disaster Relief". In: *International AAAI Conference on Weblogs and Social Media (ICWSM)*.

Landwehr, P. M. and Carley, K. M. (Jan. 2014). "Social Media in Disaster Relief - Usage Patterns, Data Mining Tools, and Current Research Directions". In: *Data Mining and Knowledge Discovery for Big Data*. Ed. by W. W. Chu. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 225–257.

Nguyen, D. T., Joty, S. R., Imran, M., Sajjad, H., and Mitra, P. (Oct. 2016). "Applications of Online Deep Learning for Crisis Response Using Social Media Information". In: *International Workshop on Social Web for Disaster Management (SWDM)*. Los Angeles, CA, USA.

Nguyen, T. H. and Grishman, R. (July 2015). "Event Detection and Domain Adaptation with Convolutional Neural Networks". In: *53rd Annual Meeting of the Association for Computational Linguistics (ACL)*. Beijing, China.

Niles, M. T., Emery, B. F., Reagan, A. J., Dodds, P. S., and Danforth, C. M. (Oct. 2018). *Social media usage patterns during natural hazards*. eprint: arXiv:1806.07451.

Olteanu, A., Castillo, C., Diaz, F., and Vieweg, S. (June 2014). "CrisisLex: A Lexicon for Collecting and Filtering Microblogged Communications in Crises". In: *AAAI Conference on Weblogs and Social Media (ICWSM)*. Ann Arbor, MI, USA.

Olteanu, A., Vieweg, S., and Castillo, C. (Mar. 2015). "What to Expect When the Unexpected Happens: Social Media Communications Across Crises". In: *Conference on Computer Supported Cooperative Work and Social Computing (ACM CSCW)*. Vancouver, BC, Canada.

Rogstadius, J., Vukovic, M., Teixeira, C. A., Kostakos, V., Karapanos, E., and Laredo, J. A. (Sept. 2013). "CrisisTracker: Crowdsourced social media curation for disaster awareness". In: *IBM Journal of Research and Development* 57.5, 4:1–4:13.

Sloan, L., Morgan, J., Housley, W., Williams, M., Edwards, A., Burnap, P., and Rana, O. (Aug. 2013). "Knowing the Tweeters: Deriving Sociologically Relevant Demographics from Twitter". In: *Sociological Research Online* 18.3, p. 7.

Snell, J., Swersky, K., and Zemel, R. S. (Dec. 2017). "Prototypical Networks for Few-shot Learning". In: *International Conference on Neural Information Processing Systems (NIPS)*. Long Beach, CA, USA.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. (Dec. 2016). "Matching Networks for One Shot Learning". In: *International Conference on Neural Information Processing Systems (NIPS)*. Barcelona, Spain.

Yan, L., Zheng, Y., and Cao, J. (Nov. 2018). "Few-shot learning for short text classification". In: *Multimedia Tools and Applications* 77.22.