TREC-IS V2 Metrics

Richard Mccreadie

May 2019

1 Executive Summary

This document summarizes the changes to the metrics between the 2018 and 2019-A versions of the Incident Streams track along with the logic for the changes.

2 Summary of lessons from 2018

The first and most significant change between the 2018 and 2019-A edition is what participants systems produce in terms of categories. In particular, participant systems are now to provide all categories that they consider to be relevant for a tweet, rather than only one. This is a direct result of our experiences from the first year, as most tweets belong to multiple categories and hence systems need to account for this. From a metrics perspective, this mandates a change to the metric set, as the 2018 'Any-type' metrics are no-longer relevant. Instead, what previously were referred to as the 'Multi-type' metrics become the default for evaluating information categorization performance.

Second, in 2018 we used classical precision and recall based metrics over all events and information categories. The conclusion from this was that these metrics were insufficient as used, since they were heavily biased towards effectiveness on the common categories (which are often not actionable ones from a response officer's perspective). As a result, it would be desirable to distinguish between performance on 'actionable' and 'non-actionable' information types.

Third, externally to the track, developers at the University of Glasgow built a test system for visualizing the output of participant systems for emergency response officers. As a result of building this system, it became clear that there were in fact two 'use-cases' for the current formulation of the task, which we refer to as the 'Alerting' use-case and the 'Information Feed' use-case. The alerting use-case represents the case where a response officer wants to receive an alert when a critical and actionable piece of information is identified. This use-case requires effective information prioritization, first and foremost, as missing an crucial call for help is a major failure. The second use-case represents the response officer subscribing to feeds of different types. In this case, the information type categorization performance is the most important.

3 Factors to Consider in Metric Development

The main factor that we care about is whether systems are able to identify and serve actionable information. There are two ways we might define actionable. First, in terms of high priority information, usually the 'high' and 'critical' levels as defined by our human assessors. Second, in terms of information categories, e.g. rescue requests are typically more important than news reports. When considering the alerting use-case, it is very important to account for high priority information. Meanwhile, for the information feed use-case, it is important to distinguish systems that do well on 'actionable' information types from those that do well on general ('non-actionable') information.

Another factor to consider is how to treat individual events. In a normal IR evaluation setting, we would normally average across 'topics', which would be equivalent to our events here. However, some events may have no actionable information at all. So, does it make sense to average over events? We would argue not, at least when considering overall system performance. It would be valuable to provide per-event metrics for further analysis however.

Third, we care about how much 'junk' we show to the user. The more junk we show the less the user will trust the system. Hence, we need a user model for using the system. For the information feed use-case, classification accuracy does to some extent capture this. However, for the alerting use-case, we are actively interrupting what the user was doing with an alert, so we need to capture that the 'cost' of an incorrect alert may escalate if we generate too many false alerts. That will likely need a greedy algorithm that simulates the number of consecutive 'irrelevant' or miss-classified posts the user sees.

4 Alerting Use-Case Metrics

For the alerting use-case in particular, we introduce new metric set to capture effectiveness for the scenario, which we refer to as Accumulated Alert Worth, and one of its components that only considers high/critical priority tweets. When calculating system performance, we will report AAW and its highPriorityWorth component.

4.1 Definitions

- t: A tweet.
- T: The set of all tweets in the test events
- $T_{high/critical}$: The set of tweets with priority of High or Critical (as judged by the assessor)
- p_t^s: priority score of the tweet given by the system
- ActC_t^s: actionable categories assigned to a tweet by the system

- ActC_t^a: actionable categories assigned to a tweet by the assessor
- NActC_t^s: non-actionable categories assigned to a tweet by the system
- NActC_t^a: non-actionable categories assigned to a tweet by the assessor
- λ: Actionable vs. nonactionable category weighting. Default=0.75
- α: Static value for a correct alert regardless of whether the categories are correct. Default=0.3. Setting this to 1.0 will cause information type categorization to be ignored.
- δ: The number of false alerts since the last true alert. Each time a tweet that is not in $T_{high/critical}$ is given a p_t^s score >= 0.7 we count that as a false alert. δ is reset to 0 each time a tweet in $T_{high/critical}$ is given a p_t^s score >= 0.7 by the system (a true alert). This is used to emulate user trust in the system over time.

4.2 Properties of Accumulated Alert Worth

AAW has the following properties:

1. The score for a system ranges between -1 and 1.
2. Only posts that the assessors judged as having 'High' or 'Critical' importance can contribute to a positive score, this means that it should be easier to end up with a negative score than a positive one, as most posts are not of high or critical importance.
3. It will penalize systems heavily if they do not get the priority level correct, as a tweet would contribute a negative score in all other cases (represents post that should have triggered an alert but did not, or generated a false alert).
4. It uses weighted information type overlap for a tweet between the system/assessor for scoring a true alert. As a result systems that return more categories more than what the assessors selected will be penalized.
5. Correctly generating a true alert will contribute at least some positive value to the system, even if the categories are wrong (see α).
6. Missing important information results in a harsh penalty (-1 score for that tweet), while producing false alerts are subject to an escalating penalty based on how many consecutive false alerts have been produced since the last true alert.
7. Events with many high or critical importance tweets will have a disproportionate impact on system performance, as we do not macro-average across events.

4.3 Scoring tweets that should generate alerts

$$\text{highPriorityWorth}(t) = \begin{cases} \alpha + ((1-\alpha)\cdot(\text{ActCScore(t)} + \text{NActCScore(t)})), & \text{if } p_t^s >= 0.7 \\ -1, & \text{otherwise} \end{cases} \tag{1}$$

$$\text{ActCScore(t)} = \gamma \cdot \frac{|\text{ActC}_t^s \cap \text{ActC}_t^a|}{|\text{ActC}_t^s \cup \text{ActC}_t^a|} \tag{2}$$

$$\text{NActCScore(t)} = (1-\gamma) \cdot \frac{|\text{NActC}_t^s \cap \text{NActC}_t^a|}{|\text{NActC}_t^s \cup \text{NActC}_t^a|} \tag{3}$$

$$\gamma = \begin{cases} \lambda, & \text{if } |\text{ActC}_t^a| > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

4.4 Scoring tweets that should not generate alerts

$$\text{lowPriorityWorth}(t) = \begin{cases} \text{argmax}\left(-\log\left(\frac{\delta}{2}+1\right), -1\right), & \text{if } p_t^s >= 0.7 \\ \text{ActCScore(t)} + \text{NActCScore(t)}, & \text{otherwise} \end{cases} \tag{5}$$

4.5 AAW Formulation

$$AAW = \frac{1}{2}\sum_{t\in T} \begin{cases} \frac{1}{|T_{high/critical}|} \cdot \text{highPriorityWorth}(t), & \text{if } t \in T_{high/critical} \\ \frac{1}{|T_{low/medium}|} \cdot \text{lowPriorityWorth}(t), & \text{otherwise} \end{cases} \tag{6}$$

5 Information Feed Use-Case Metrics

For the Information Feed use-case, we want to evaluate the quality of a TREC-IS system in terms of how effectively it can categorize the tweets into the 25 high-level information types in the TREC-IS Ontology. Furthermore, we want to distinguish between actionable and non-actionable information types. In this case, we will use the same Multi-type classification metrics that we used in 2018. The difference is that we also report performance over six actionable types, 'Request-GoodsServices', 'Request-SearchAndRescue', 'CallToAction-MovePeople', 'Report-EmergingThreats', 'Report-NewSubEvent', 'Report-ServiceAvailable' as well. Performance is reported in terms of classification F1 on the positive tweets (i.e. true positives and false positives). This intentionally ignores false negatives and true negatives, as we are more interested in the good information reaching the feed, rather than noise being added. However, to capture noise we do report overall classification accuracy.

6 Re-generated 2018 Results under Script V2

Below we have re-generated the performances for the TREC Incident Streams 2018 participating systems under the updated metrics for reference. Note that the evaluation script has been updated to 'V2', so if evaluating your own systems check that you are using the updated script from the website.

Run	Alerting		Information Feed			Prioritization	
	Accumulated Alert Worth		Info. Type Positive F1		Info. Type Accuracy	Priority RMSE	
	High Priority	All	Actionable	All	All	Actionable	All
uogTr.R1.asp	-0.5645	-0.0830	0.0148	0.0793	0.8758	0.1286	0.0931
uogTr.R2.asp	-0.5974	-0.0840	0.0078	0.1075	0.8811	0.1179	0.0927
uogTr.R3.asp	-0.5846	-0.0831	0.0078	0.1086	0.8938	0.1184	0.0912
cbnuC1	-0.7983	-0.0573	0.0000	0.1062	0.8880	0.1751	0.0940
cbnuC2	-0.7983	-0.0573	0.0000	0.1105	0.8890	0.1751	0.0941
cbnuS1	-0.7983	-0.0573	0.0136	0.1123	0.8924	0.1751	0.0940
cbnuS2	-0.7983	-0.0573	0.0392	0.1247	0.8934	0.1751	0.0940
DLR_Augmented	**-0.1618**	-0.0759	0.0347	**0.1359**	0.8862	0.0835	0.1601
DLR_Baseline	-0.3371	-0.0762	0.0000	0.0763	0.8935	0.1064	0.1495
DLR_Fusion	-0.1554	-0.0770	**0.0536**	0.1262	0.8853	0.0831	0.1597
DLR_Simple_CNN	-0.1641	-0.0757	0.0204	0.1207	0.8917	0.0824	0.1643
IITBHU1	-0.5828	-0.0758	0.0299	0.1074	0.8761	0.1533	0.1246
IITBHU2	-0.5828	-0.0758	0.0299	0.1074	0.8761	0.1533	0.1246
KDEIS1_CLSTM	-0.6990	-0.0712	0.0000	0.0632	0.8741	0.1173	0.0841
KDEIS2_ACBLSTM	-0.6990	-0.0712	0.0000	0.0517	0.8659	0.1166	0.0840
KDEIS3_ACSBLSTM	-0.6990	-0.0712	0.0000	0.0665	**0.9032**	0.1173	0.0841
myrun1	-0.7869	-0.0581	0.0287	0.0725	0.8728	0.1180	0.0726
myrun2	-0.7869	-0.0582	0.0287	0.0787	0.8737	0.1363	0.0790
myrun-10	-0.7869	-0.0587	0.0000	0.0730	0.8984	0.1352	0.0941
myrun-11	-0.7869	-0.0586	0.0000	0.0712	0.8978	0.1358	0.0939
myrun-21	-0.6684	-0.0704	0.0074	0.0944	0.8909	0.1142	0.0954
NHK$_r un1$	-0.7086	-0.0540	0.0339	0.1052	0.8867	**0.0733**	**0.0599**
NHK$_r un2$	-0.7370	-0.0548	0.0227	0.1129	0.8941	0.0910	0.0621
NHK$_r un3$	-0.7123	**-0.0531**	0.0000	0.1091	0.8957	0.0897	0.0610
NHK$_r un4$	-0.7763	-0.0568	0.0000	0.0841	0.8904	0.1159	0.0661
SINAI$_r un1$	-0.7979	-0.0631	0.0000	0.0698	0.8916	0.1604	0.0902
SINAI$_r un2$	-0.7892	-0.0599	0.0000	0.0703	0.8833	0.1508	0.0855
SINAI$_r un3$	-0.7974	-0.0607	0.0000	0.0736	0.8906	0.1602	0.0891
SINAI$_r un4$	-0.7974	-0.0603	0.0000	0.0760	0.8872	0.1670	0.0923
umdhcilbaseline	-0.7983	-0.0573	0.0284	0.0843	0.8802	0.1751	0.0940
umdhcilfasttext	-0.7983	-0.0573	0.0074	0.1113	0.8843	0.1751	0.0940
umdhcilfts	-0.7983	-0.0573	0.0072	0.0400	0.8896	0.1751	0.0940
umdhcilspread	-0.7983	-0.0573	0.0000	0.0819	0.8921	0.1751	0.0940
UPB$_D ICE1$	-0.7461	-0.0717	0.0081	0.0517	0.8738	0.1261	0.0912
UPB$_D ICE2$	-0.7795	-0.0600	0.0000	0.0522	0.8863	0.1592	0.0922
UPB$_D ICE3$	-0.7983	-0.0573	0.0000	0.0398	0.8724	0.1748	0.0939
UPB$_D ICE4$	-0.7979	-0.0573	0.0000	0.0437	0.8806	0.1630	0.0874

Table 1: 2018 Submitted runs under the v2 evaluation script