

# Automated Pop-Up Fact-Checking: Challenges & Progress

Bill Adair,<sup>†</sup> Chengkai Li,<sup>‡</sup> Jun Yang,<sup>†</sup> Cong Yu<sup>§</sup>

<sup>†</sup>Duke University, <sup>‡</sup>University of Texas at Arlington, <sup>§</sup>Google Research

## ABSTRACT

We present a system for automated “pop-up” fact-checking in real time, which we are building as part of the *Tech & Check Cooperative*. This system aims at delivering relevant fact-checks to users right at the time when they encounter false claims. We provide an overview of the system architecture, describe the key technical challenges, present our current progress, and outline several future directions.

## 1 Introduction

Today, we are struggling with an unprecedented amount of falsehoods, hyperboles, and half-truths that do harm to democracy, health, economy, and national security. Fact-checking is a vital tool for defending against this onslaught, perhaps now more than ever. Despite the rise of fact-checking efforts globally, fact-checkers find themselves increasingly overwhelmed and their messages difficult to reach some segments of the public. The *Tech & Check Cooperative* is a project at the Duke Reporters’ Lab that brings together many collaborators and partners at Duke, UT-Arlington, and Google Research. An overarching goal of this project is to leverage the power of data and computing to help make fact-checking and dissemination of fact-checks to the public more effective, scalable, and sustainable.

Under this project, we have been building a system to support “pop-up fact-checking,” a vision that we had laid out in the 2017 Computation+Journalism Symposium [1]. A reason for building this system is that most people do not take the time to research what they see or hear; even if they do, they may do so selectively, skipping those claims that confirm their prior beliefs. Although a good number of fact-checks already exist, they are not reaching their audience. Pop-up fact-checking would deliver fact-checks to people right at the time when they consume misinformation (without relying on them to seek out fact-checks themselves).

We believe that the time for pop-up fact-checking has come because of a confluence of several factors. First, most claims are not stated just once; they are repeated even after being refuted. For example, in the United States, both major political parties typically have “talking points” containing claims that are repeated by many politicians through various outlets over time. Thus, there are plenty of opportunities to catch repeated claims.

Second, we also now have a large and growing database of up-to-date fact-checks with which to catch repeated claims, thanks to *ClaimReview*. *ClaimReview* is a schema.org standard for “marking up” published fact-checks, allowing search engines and other automated platforms to locate and interpret fact-checking contents easily. Using *ClaimReview*, we have created *Share the Facts*, a database of fact-checks by reputable fact-checking organizations, as a primary resource for matching claims. Even when a fact-checked claim is made by a different person, that fact-check can have relevance as a pop-up during a live event.

Finally, devices such as smartphones and smart TVs have become ubiquitous. With these devices, it is possible to know what a user is experiencing, and augment this experience with fact-checks in real time. Imagine that you turn on your TV to watch a debate. When a candidate makes a claim that has been fact-checked before, a box would appear on the screen telling you about a related fact-check. The system could work on newer smart TVs and on video-streaming websites. People with older TVs can use a smartphone app to get a “second-screen” experience that streams fact-checks synchronized with the live event.

In the remainder of this paper, we first describe a basic automation pipeline for implementing this system, and then discuss various challenges that arise. We present our progress and some ideas for coping with these challenges.

## 2 A Basic Automation Pipeline

Figure 1 shows our basic automation pipeline to support pop-up fact-checking. On a high level, it takes the audio/video stream of a live event as input, converts it to a stream of text, filters it to find checkable and check-worthy claims, and then matches these against our database of fact-checks to produce a stream of fact-checks. We now describe each of these steps in the pipeline in some more details below. In later sections, we will revise this pipeline to address additional challenges we face.

**Converting speech to text.** For this step of the pipeline, we have experimented with various methods. The UT-Arlington team used a device to extract the close captions of the TV broadcasting programs. Experiments with the live debates during the 2016 U.S. presidential election revealed that the live closed captions themselves were oftentimes inaccurate; the identities of speakers were not always available, which made it difficult to apply

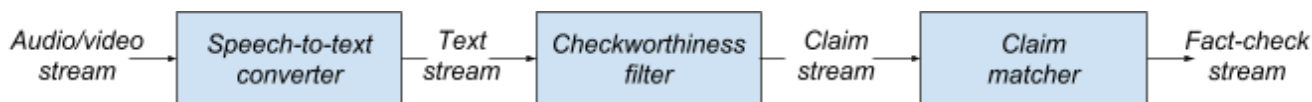


Figure 1. A basic automation pipeline for pop-up fact checking.

contextual and background information. The Duke team used the Google Cloud Speech-to-Text API to transcribe audio. This API returns a confidence score for each of its transcribed sentences; we discarded those lower than 0.9 confidence. As example, there were about 400 sentences in President’s Trump’s 2018 State of the Union address. The API was perfect on 75% of them and near perfect on 15% of them, but the remaining results were unusable. In the end, this step produced about 370 well-transcribed sentences for the speech. The implementation had to address a number of technical issues, such as a limitation of the API that required some buffering of the audio, and the fact that a transcription may be “revised” upon receiving more input. Generally speaking, there is a tradeoff between accuracy and timeliness: correct transcription may require holding off until sufficient context and redundancy have been built up. Our implementation added a delay of 30 seconds. This tradeoff could be more pronounced for more complex tasks, such as speaker identification, coreference resolution, and entity recognition and disambiguation; further study is still needed.

**Checkworthiness filter.** For this step, we used *ClaimBuster* [2-5] developed by the UT-Arlington team. ClaimBuster gives each sentence a score to indicate how likely it contains a factual claim worth checking. Given the plethora of claims we are constantly exposed to, this functionality helps fact-checkers focus efficiently on the most checkworthy sentences without painstakingly sifting through them all. The machine learning model used by *ClaimBuster* was trained using human-labeled sentences from past presidential debates and it was tested in real-time during the live coverage of all primary election and general election debates for the 2016 election. Since then, it has also been used in finding important factual claims from public figures’ tweets. The ClaimBuster API has been also used in a sister project in the Tech & Check Cooperative to provide an “alerts” service that brings checkworthy claims to the attention of fact-checkers, by automatically scraping transcripts from websites, parsing the sentences, and submitting them to the ClaimBuster API. In the context of pop-up fact-checking, however, we simply use the ClaimBuster API as a checkworthiness filter on the text stream, to obtain a smaller stream of potential claims to be further processed by the next step of our pipeline (claim matching). We used a fairly low ClaimBuster score threshold (0.25) for higher recall, because precision can be further improved with claim matching. For example, for the 2018 State of the Union address, about 200 sentences passed this filter and were sent to the next step.

**Claim matching.** For each sentence that passed the checkworthiness filter, we match the claim to the database of fact-checks to find if any existing fact-checks can help assess the veracity of the claim. Several variants of the problem are possible. First, it can be considered as a traditional information retrieval problem, not unlike Web search (but restricted to the database of fact-checks), where we aim to return fact-checks “most relevant” to this claim. The user would be able to infer the veracity of the claim by examining the top results, although this inference must be carried out by the user. If there are no perfect match or there are many related fact-checks, the user can still explore the results for future investigation.

The second variant of the problem goes further: we compute a verdict on a given claim algorithmically. This variant would make the results more direct and easier for real-time consumption, and could also be used to support voice-based apps such as Alexa’s *Ask the Fact Checkers* [6]. This problem can be framed, at least in part, as the *entailment* problem in NLP (natural language processing). The database of fact-checks can be seen as a collection of assertions of the form “ $X$  is true” or “ $X$  is false.” Given a claim  $Y$ , the question is whether the database entails either “ $Y$  is true” or “ $Y$  is false.” Both  $X$  and  $Y$  are stated in natural language. In general, the truth value of  $Y$  may follow from multiple assertions in the database, although the entailment problem in NLP typically assumes one  $X$  and one  $Y$ . A further complication is that in many cases it is difficult to assign a true or false verdict to a claim, e.g., because the claim is ambiguous, or because the claim is factually correct but still misleading [7, 8]. Nonetheless, even a limited entailment test for a pair of sentences could go a long way in offering a practical solution.

We have investigated multiple approaches to solving the above problems. In [9], the Google team explored a related problem of given a fact-checking article, identifying all articles on the Web that are either spreading or confirming the claim made within the fact-checking article. The solution involves three stages. In the first stage, we generate search queries based on the content of the fact-checking article as well as the claim text embedded within ClaimReview. Those queries are issued against Google and we considered the topic returned results as related documents. In the second stage, *relevance classification*, we built a machine learning model to identify documents that are “relevant”: namely, those that are indeed talking about the given claim, instead being merely talking about the same topics as the claim. To build this model, we leveraged features such as entity similarity and sentence similarity between the candidate document and the

fact-checking article. In the last stage, *stance detection*, we built an machine learning model to predict [contradict, support] label for the given pair of candidate document and fact-checking article. The unique feature employed by this model is the vocabulary for contradiction, i.e., phrases that strongly indicate disagreements. Experiments showed that our model outperforms state-of-art baselines for both relevance classification and stance detection.

The UT-Arlington team examined the efficacy of using textual entailment algorithms for claim matching. Specifically, we conducted experiments using the Excitement Open Platform (EOP) [10], which implements major state-of-the-art entailment algorithms as well as linguistic analyses and knowledge resources. While there are datasets for evaluating general-purpose entailment algorithms, there is no such a dataset specifically for claim matching. Therefore we also decided to create a benchmark dataset. The dataset consists of sentence pairs formed by factual claims from the 2016 primary and general election debate transcripts and PolitiFact fact-checked claims. We manually annotated these pairs, which turned out to be comprised of 12.5% entailment pairs and 87.5% non-entailment pairs. We trained the EOP algorithms on their provided dataset, and then we evaluated the algorithms' performance on our labelled dataset. The Max Entropy classification algorithm coupled with OpenNLP had the highest recall (99%) for non-entailment pairs, while the Edit Distance PSO algorithm attained the highest recall (90%) for entailment pairs. The 99% recall is surprisingly high, and we understand that it is partly due to the unbalanced test data.

The Duke team developed an in-house relevance ranking algorithm and experimented with an adaptation of a model developed by OpenAI and released in June 2018 [11]. This model attempts to solve two problems which have plagued traditional natural language inference approaches: the difficulty of identifying word dependencies, and the limited quantity of labeled data. The first problem was addressed by the "Transformer" framework, a type of neural network proposed in December 2017 [12]; it improves upon previous neural network architectures through use of "attention mechanisms," which determine meaning through a weighted average of dependency on all other words in a sentence. The second problem was addressed by an unsupervised preprocessing step. OpenAI used a large corpus to "teach" the Transformer English, and with minimal adaptation was able to transfer this unsupervised learning to beat the state-of-the-art results on common NLP datasets. Using OpenAI's published weights from the unsupervised learning step, we were able to achieve near state-of-the-art results on the standard Multi-Genre NLI dataset (MNLI). If the model did not attempt to distinguish between entailment and contradiction, the accuracy would increase to 85.8%. On the other hand, for the 2018 State of the Union address, this more sophisticated approach did not yield better results than our simpler in-house relevance ranking algorithm (which returned 30 fact-checks for the speech). The good performance of the simpler algorithm in this case can be

explained by the fact that this event was widely fact-checked; furthermore, our database had fact-checks that specifically targeted this event, done by fact-checkers after the event, so the wordings were exact matches (and hence easy to detect) for many results. For a new event, we would not expect the simpler algorithm to perform as well. More evaluation efforts are ongoing.

### 3 Accuracy Challenges

No algorithm in our automation pipeline is perfect. Accuracy challenges for speech-to-text conversion and ClaimBuster have been well documented in the literature, so we shall focus mostly on claim matching here. First, there are fundamental challenges in understanding natural languages, many of which we had no workable solutions for until recently. A commonly used example in the field of NLP is the pair of sentences, "The animal didn't cross the street because it was too wide" and "the animal didn't cross the street because it was too tired." How would an algorithm decide whether "it" refers to the animal or the street? As another example, related to fact-checking, consider two questions that differ in only one word: "Is it true when  $A$  said  $X$ ?" and "is it true that  $A$  said  $X$ ?" For the first question, we are more concerned with checking the claim  $X$ ; for the second, however, we would be more concerned with checking whether  $A$  ever made that claim.

Second, beyond ambiguities that arise at the language level, in many cases there are nuances in what exactly is being checked. For example, in January 2018, two fact-checking organizations gave seemingly different ratings ("mostly true" [13] vs. "spins the facts" [14]) on President Trump's claim that black unemployment rate was the lowest level recorded. Upon closer examination, we can see that the difference stems from whether one focuses on verifying the factual correctness of the claim on unemployment, or checking how that claim was used in an argument (that it was because of the President's policies). Today, algorithms still seem nowhere near in recognizing such nuances in reasoning. Sometimes a claim may be made without explicitly stating what argument it supports, yet the suggestion is clear to the audience. Sometimes a set of claims may be carefully woven together into arguments, but with subtle logical fallacies. These issues make it very difficult to make a verdict, let alone automatically.

Third, being able to match claims to fact-checks accurately requires a lot of contextual information and domain knowledge. Some of the ambiguity in natural language cannot be resolved without domain knowledge: e.g., what are the reasonable definitions of "small business owners"? Even the simplest form of contextual information, e.g, speaker and time, is crucial: two sentences identically worded can be two very different claims when stated by different persons or at different times. Recent advances in natural language processing still need to be extended with more consideration for contextual information and domain knowledge, beyond what the sentences themselves have to offer.

What are doing about these challenges then? We outline three directions we are currently pursuing below.

**Human-in-the-loop curation.** Recognizing that algorithms are not (and perhaps never will be) perfect, we have augmented our automation pipeline with human curation. First, for each audio/video stream, we track the intermediate results from each step of the pipeline: i.e., the text stream generated by the speech-to-text converter, the claim stream filtered by ClaimBuster, and the fact-check stream produced by the claim matcher. Since a mistake made by an earlier step will affect subsequent steps, we also track the lineage among intermediate results, so it is easy to trace an error and correct all affected results. We keep not only the “best” results produced by our algorithms, but also those that score not as high, in case that some of them turn out to be correct.

For important events, we envision that a group of human editors will monitor the event and the results from our basic automation pipeline, and make any necessary corrections in real time. The Duke Team is currently working on an interface for human curation. For some events, a delay may be introduced to ensure that human editors have a chance to vet the matching fact-checks before serving them to the public. This mode of operation is in fact similar to *FactStream*, an app developed by the Tech & Check Cooperative, which had delivered fact-checking live events before relying purely on inputs from human editors. The advantages of the new pop-up fact-checking system over purely human inputs are that it will reduce the workload and increase effectiveness of human editors, and that it will be much less dependent on their having an intimate knowledge of all existing fact-checks. These advantages can be crucial if we want to make live fact-checking possible for events at state or local levels, where it would be much more difficult to find the manpower and human expertise.

Finally, by making the transient results persistent, our system allows curation efforts (and automation) to continue beyond an live event. After the event, we can still update the associated fact stream to insert new fact-checks or make corrections, and these updates will be visible to any user viewing the event later.

**Deeper understanding of claims.** The UT-Arlington team is exploring structured and semantic modeling that can capture various aspects of a factual claim, including its domain and topic, the template of the fact being expressed, the entities involved and their relationships, quantities, time points and intervals, comparisons, and aggregate structures. With such modeling capability in place, we will be able to develop automation tools that exploit the idiosyncrasies of different forms of factual claims. For instance, the claim-matching algorithm can go beyond current methods for paraphrase detection, semantic similarity, and textual entailment, and apply direct, fine-grained comparison of claims’ structured representations.

Our latest progress along this direction is an extension of the Berkeley FrameNet project [15], a lexical resource for English built on the theory of meaning called frame semantics [16]. Our extension of FrameNet includes 13 new frames specifically tailored to factual claims. Each frame comes with a definition and descriptions of its elements as well as a set of example sentences which are annotated with the corresponding frame elements. For automatically identifying frames from text and further identifying the frames’ elements, we used Open Sesame [17], an open-source frame identification and frame element extraction tool based on recurrent neural networks. For training its machine learning models, we annotated 900 sentences which were gathered from fact-checks released by PolitiFact in the past. The trained models can be applied on any text to detect sentences that fall within the scope of our frames as well as constituent elements of frames.

**Labeled data procurement.** As can be seen in earlier discussion, many of our algorithms are based on machine learning and require lots of labeled data for training and evaluation. While standard benchmark datasets such as MNLI do exist, they tend to be broad and do not have enough examples to expose the various subtleties that arise specifically in matching fact-checks. Aside from the data labeling efforts already described above, in an effort to procure more training data for claim matching, the Duke team developed a simple crowdsourcing tool that asks users to paraphrase claims that have been fact-checked in Share the Facts database. As it turned out, however, most users were not very “creative” in their paraphrasing, so the additional training data did not result in significant improvement in accuracy. We are now investigating two other approaches: one is to look on the Web and social media for instances where claims have been made in different forms; the other is to use gamification to make it interesting for users to create less trivial training examples.

## 4 Scalability Challenges

For a widely viewed live event, it is clear that we cannot to afford to run an instance of our automation pipeline for every viewer. Instead, for scalability, we run this pipeline only once, and produce the (curated) fact-check stream in real time so that end-user app can simply “tune in” to the appropriate fact-check stream. Then, scaling up the serving of this fact-check stream is much easier. We also plan to work on having a smartphone app automatically identify the audio stream that it is listening to, and more importantly, the time into that stream, so that it can serve the correct fact-check stream in a synchronized manner.

As we employ more sophisticated algorithms, their running time also becomes an issue as we aim to provide fact-checking in real time. We need to cap the latency introduced by each step of the automation pipeline such that it can keep up with a live event. Aside from speech-to-text and human curation delays, the step dominating the end-to-end latency is claim matching. For

example, our current (not very optimized) implementation of the Transformer-based matching algorithm takes a few seconds to process a pair of a claim and a fact-check on a laptop. Fortunately, the algorithm is amenable to GPU parallelization, and matching against multiple fact-checks is just embarrassingly parallel. Also, a faster, cruder matching algorithm can be used to filter the set of candidate matches to be further examined by this more expensive algorithm. Therefore, we believe that real-time claim matching is well within our reach even with the most sophisticated algorithms.

## 5 User Interface Challenges

In addition to the computational hurdles, pop-up fact-checking presents new challenges for user interface and user experience design. Questions include: What are the most effective ways for people to interact with fact-checking content on TV and/or mobile devices? Should we display ratings such as “Truth-O-Meters” and “Pinocchios” on these devices? The Duke Reporters’ Lab collaborated with Blink, a firm that specializes in UX testing, to conduct a small experiment in October 2018. Excerpts from two State of the Union addresses, one featuring Obama and one with Trump, were matched with actual fact-checks published by PolitiFact. The fact-checks were presented in different formats on a large-screen TV at the appropriate times during the speeches. Fifteen people were recruited to watch the speeches and then were interviewed about their reactions to the on-screen fact-checking.

The results are still being analyzed as this paper is being written, but the preliminary data indicates strong support for on-screen fact-checking. Participants said they liked the information from the fact-checks and they understood the label “Related Fact Check,” which indicated the information presented was not necessarily directly about the precise claim made in the speech. There also was a strong preference to have the information in the lower third of the screen rather than in the upper or lower corners. There was less consensus about whether the information should include rating words or devices indicating whether a claim was true, or false, etc. Eight of the 15 participants preferred just text without such a rating; seven preferred a rating word or an image of Truth-O-Meter. The results pointed to new avenues for further user experience research that we plan to conduct in the future.

## ACKNOWLEDGMENTS

*This work was support by NSF grants #1408846, #1408928, #1719054 and #1718398; grants from the Knight Foundation, the Facebook Journalism Project, and the Craig Newmark Foundation; the Bass Connections and Data+ programs at Duke University; and Google Inc.*

*The authors would also like to acknowledge their collaborators Pankaj K. Agarwal (Duke) and Mark Tremayne (UT Arlington), the teams of students and staff working on the project at Duke (Naman Agarwal, Archana Ahlawat, Towqir Aziz, Xinghao Cheng, Fangge Deng, Lucas Fagan, Sherry Feng, Christopher Guess, Ethan Holland, Mayuresh Kunjir, Drew Learner, Junbo Li, Helena Merk, Matthew O’Boyle, Scott Heng On, Asa*

*Royal, Erica Ryan, Mark Stencel, Caroline Wang, Wenqin Wang, Jason Wang, Yuhao Wen, Connie Wu, Harry Xie, Liuyi Zhu) and at UT Arlington (Priyank Arora, Fatma Arslan, Sameer Chaulagain, Jacob Devasier, Kyrell Dixon, Damian Jimenez, Sumeet Lubal, Sarthak Majithia, Daniel Obembe, Shadetur Rahman, Sarbajit Roy, Theodora Toutountzi), as well as Erica Anderson, Simon Baumgartner, Lee Butler, Corinna Cortes, Michelle Danoff, Flip Korn, Justin Kosslyn, Lisa Lehman, Riley Matthews, Danny Poritz, Jane Shapiro, Maggie Shiels, Xuezhi Wang, Clement Wolf at Google.*

## REFERENCES

- [1] Bill Adair, Chengkai Li, Jun Yang, and Cong Yu. “Progress toward ‘the holy grail’: the continued quest to automate fact-checking.” In *Proc. of the 2017 Computation+Journalism Symposium*, Evanston, Illinois, USA, October 2017.
- [2] Naeemul Hassan, Bill Adair, James T. Hamilton, Chengkai Li, Mark Tremayne, Jun Yang, and Cong Yu. “The quest to automate fact-checking.” In *Proc. of the 2015 Computation+Journalism Symposium*, 2015.
- [3] Naeemul Hassan, Fatma Arslan, Chengkai Li, and Mark Tremayne. “Toward automated fact-checking: Detecting check-worthy factual claims by claimbuster.” *KDD*, pages 1803–1812, 2017.
- [4] Naeemul Hassan, Gensheng Zhang, Fatma Arslan, Josue Caraballo, Damian Jimenez, Siddhant Gawsane, Shohedul Hasan, Minumol Joseph, Aaditya Kulkarni, Anil Kumar Nayak, Vikas Sable, Chengkai Li, and Mark Tremayne. “Claimbuster: The first-ever end-to-end fact-checking system.” *PVLDB*, demonstration description, 10(12):1945–1948, 2017.
- [5] Damian Jimenez and Chengkai Li. “An empirical study on identifying sentences with salient factual statements.” *IJCNN*, 8 pages, 2018.
- [6] <https://reporterslab.org/fact-checking-comes-amazon-echo/>
- [7] You Wu, Pankaj K. Agarwal, Chengkai Li, Jun Yang, and Cong Yu. “Computational fact checking through query perturbations.” *ACM Transactions on Database Systems*, 42(1):4:1-4:41, March 2017.
- [8] Jun Yang, Pankaj K. Agarwal, Sudeepa Roy, Brett Walenz, You Wu, Cong Yu, Chengkai Li. “Query Perturbation Analysis: An Adventure of Database Researchers in Fact-Checking.” *IEEE Data Eng. Bull.* 41(3): 28-42 (2018).
- [9] Xuezhi Wang, Cong Yu, Simon Baumgartner, Flip Korn. “Relevant Document Discovery for Fact-Checking Articles.” *The Web Conference (Journalism Track)*, 2018.
- [10] Bernardo Magnini, Roberto Zanolini, Ido Dagan, Kathrin Eichler, Guenter Neumann, Tae-Gil Noh, Sebastian Padó, Asher Stern, Omer Levy. “The Excitement Open Platform for Textual Inferences.” *ACL (System Demonstrations)*, pages 43-48, 2014.
- [11] <https://blog.openai.com/language-unsupervised/>
- [12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin. “Attention is All you Need.” *NIPS*, 2017.
- [13] <https://www.politifact.com/truth-o-meter/statements/2018/jan/08/donald-trump/how-accurate-donald-trumps-about-black-hispa/>
- [14] <https://www.factcheck.org/2018/01/trump-takes-undue-credit-black-unemployment/>
- [15] <https://framenet.icsi.berkeley.edu/fndrupal>
- [16] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. “The Berkeley FrameNet Project.” *ACL*, pages 86–90, 1998.
- [17] <https://github.com/swabhs/open-sesame>



