WikiCheck: An End-to-end Open Source Automatic Fact-Checking API based on Wikipedia

Mykola Trokhymovych
Ukrainian Catholic University
trokhmovych@ucu.edu.ua

Diego Saez Trumper
Wikimedia Foundation
diego@wikimedia.org
Agenda

1. Introduction
2. Related work
3. Data observation
4. System architecture
5. Experiments
6. Demo
7. Summary
Introduction. Motivation

- False facts are influential
- Manual fact-checking is time-consuming
- Automation reduces time to "stick" in minds.

Third of Russians think sun spins round Earth?

source: Reuters

Disinformation example:

source: cnbc (2013)

Disinformation influence:

Temporary loss of market cap in the S&P 500 alone totaled $136.5 billion

source: Radio Liberty

source: Reuters

Fallout Over Flat-Earth Theory Hits Russia's 'Emmy' TV Awards
Why Wikipedia?

- Using traceable information, coming from reliable sources
- One the most extensive open knowledge bases in the world
- Can be used as evidence source for facts validation
- Not perfect data source, but tends to be :)

source: SimilarWeb
Introduction. Problem formulation

End-to-end fact-checking:

Given the **claim**, classify it as true or false and provide **evidence** for your reasoning from a reliable **knowledge base**

Natural language inference (NLI):

Given two texts (**claim** and **hypothesis**), decide if the **hypothesis** supports the initial claim, refutes it, or does not relate to it.

Explanation:

**Claim:** "Today is Wednesday"

**Hypothesis (evidence):** "Tomorrow is Thursday"

**Knowledge base:** Wikipedia
Open problems

- The efficiency of NLI models is not considered in previous research
- Lack of high-quality NLI datasets for model training
- Software architecture for end-to-end fact-checking
Research goals

- Analyze NLI datasets. Define the specific data features and limitation, design a methodology for data quality improvement.
- Experiment with NER models usage for information retrieval stage.
- Build accurate and efficient domain specific sentence-based NLI model. Experiment with unsupervised learning and transfer learning.
- Implement an open-source end-to-end fact-checking API.
Related work
Masked language modeling

BERT-like models

Bidirectional Encoder Representations from Transformers (Devlin et al., 2018).

How to get sentence embeddings?

Sentence-BERT (Reimers and Gurevych, 2019)

1) CLS token
2) Mean of tokens embeddings
3) Build a model on top of token embeddings

source: jalammar.github
Natural language inference

Word-based approach

Sentence A: Today is Wednesday
Sentence B: Tomorrow is Thursday

[CLS] [SEP]

MLM Classifier Label: SUPPORTS

Sentence-based approach

Sentence A: Today is Wednesday
Sentence B: Tomorrow is Thursday

[CLS] [CLS]

MLM Classifier Label: SUPPORTS

Main previous contributions:

- Using composition of embeddings of different types. (Kiela et al., 2018)
- BiLSTM + Max Pooling for sentence embeddings for NLI. (Talman, et al., 2019)
- Using multitask learning and MLM (Liu et al., 2019) (word-based approach)
- Using semantics information for NLU (Zhang et al., 2020) (word-based approach)

Why sentence-based approach:

- Allows caching of sentence embeddings
- Allows processing claim and hypotheses separately
- Usually lighter and faster on inference
- Usually lower accuracy
Fact checking systems

**Academic works:**

FEVER: a large-scale dataset for Fact Extraction and Verification (Thorne et al., 2018b)

**General architecture:**

1. Article selection
2. Sentence selection
3. NLI model
4. Aggregation

**Industry solution:**

“Kyiv is the capital of Poland.”

We have compiled a list of related fact checks and evidence to give you some context around this claim:

**Similar Facts**

1. Kyiv is governed by fascists

**Evidence**

72%
Data observation
General information

General domain datasets

- **SNLI**: Comes from image captions. The first and the main benchmark dataset for the NLI task.

- **MNLI**: Comes from a wide range of styles, degrees of formality, and topics: conversations, reports, speeches, letters, fiction.

Specific domain datasets

- **WIKIFACTCHECK-ENGLISH**: Comes from modified Wikipedia texts. Includes context.

- **FEVER**: Manually generated and labeled claims. Related evidences as links to Wikipedia dump.
### SNLI and MNLI. Data Sample

#### Original data sample

| Dataset | Claim | Hypothesis | Label   |
|---------|-------|------------|---------|
| MNLI    | The Old One always comforted Ca’daan, except today. | Ca’daan knew the Old One very well. | neutral |
| MNLI    | At the other end of Pennsylvania Avenue, people began to line up for a White House tour. | People formed a line at the end of Pennsylvania Avenue. | entailment |
| SNLI    | A man inspects the uniform of a figure in some East Asian country. | The man is sleeping | contradiction |
| SNLI    | An older and younger man smiling. | Two men are smiling and laughing at the cats playing on the floor. | neutral |
SNLI and MNLI. Annotation artifacts

Distributions of length of hypothesis in training dataset

![Graph showing distributions of length of hypothesis in SNLI and MNLI datasets](image-url)
SNLI and MNLI. Annotation artifacts

SNLI dataset top-15 the most frequent hypothesis and their classes counts

We observe disbalance across labels of samples with the same hypothesis.
Data observation. FEVER

Original data sample

{"id": 75397,  
"verifiable": "VERIFIABLE",  
"label": "SUPPORTS",  
"claim": "Nikolaj Coster-Waldau worked with the Fox Broadcasting Company.",  
"evidence": [[[92206, 104971, "Nikolaj_Coster-Waldau", 7],  
[92206, 104971, "Fox_Broadcasting_Company", 0]]]}

FEVER data sample. Article linking.

| Claim                                      | Evidence Articles                      |
|--------------------------------------------|----------------------------------------|
| Nikolaj Coster-Waldau worked with the Fox Broadcasting Company. | Fox_Broadcasting_Company, Nikolaj_Coster-Waldau |
| Hermit crabs are arachnids.                | Arachnid, Hermit_crab, Decapoda        |
| There is a capital called Mogadishu.       | Mogadishu                               |

FEVER data sample. SNLI-style relation dataset.

| Claim                                      | Hypothesis                                                                 | Label       |
|--------------------------------------------|----------------------------------------------------------------------------|-------------|
| Roman Atwood is a content creator.         | He is best known for his vlogs, where he posts updates about his life daily. | SUPPORTS    |
| Selena recorded music.                     | Selena began recording professionally in 1982. Selena Selena (film)        | SUPPORTS    |
Negative sampling. FEVER

Original data sample:

{"id": 93826,
"verifiable": "NOT VERIFIABLE",
"label": "NOT ENOUGH INFO",
"claim": "Donna Noble is played through improv.",
"evidence": [[111196, None, None, None]]}

{"id": 75397,
"verifiable": "VERIFIABLE",
"label": "SUPPORTS",
"claim": "Nikolaj Coster-Waldau worked with the Fox Broadcasting Company.",
"evidence": [[92206, 104971, "Nikolaj_Coster-Waldau", 7],
[92206, 104971, "Fox_Broadcasting_Company", 0]]}

"Donna Noble is played through improv."

Given the original sample from SUPPORTS or REFUTES class

1) Extract "Donna Noble" named entity
2) Find the corresponding article
3) Pick the random sentence from it

1) Extract sentences from all related articles. For example from: "Nikolaj_Coster-Waldau" and "Fox_Broadcasting_Company"
2) Pick the random sentence that was not previously used for SUPPORTS or REFUTES class samples
System architecture
Application design

Model level one

- Query enhancing using NER models
- Wikipedia open search API

Top n articles (candidates for being evidence source)

Model level two

- Splitting into sentences
- NLI model. Classification model

Claim

Hypothesis

Classification result. (Correct, Incorrect, No related information)
Model level one: Validation

Example:

Query:
Charles, Prince of Wales is patron of numerous other organizations.

Ground truth pages links:
{‘Charles, Prince of Wales’}

Set of 5 pages candidates:
{‘Charles, Prince of Wales’, ‘Charles’, ‘Charles_City_County, Virginia’, ‘Grace_Kelly’, ‘Prince_Harry, Duke_of_Sussex’}

Recall: 1

Recall = \frac{true \, positives}{true \, positives + false \, negatives}
Model level two

Claim → BERT → u → Concatenating embeddings → Dense layer → Softmax

Hypothesis → BERT → v → Sentence embeddings → Dense layer → Softmax
Experiments
Improving the search

Metrics:
- Average Recall (AR)
- Average number of candidates returned.

Possible modifications:
- Use out-of-the-box NER models from SpaCy or Flair
- Strategy of treating named entities: merging or separate queries
- Increase N - number of candidates to extract for each query
## Improving the search. Results

| Configuration                  | AR (higher is better) | N returned, (lower is better) |
|-------------------------------|-----------------------|-------------------------------|
| No NER model N=10             | 0.628                 | 9.11                          |
| No NER model N=30             | 0.645                 | 25.02                         |
| No NER model N=50             | 0.649                 | 39.16                         |
| SpaCy sm merged N=10          | 0.810                 | 15.33                         |
| SpaCy sm merged N=30          | 0.833                 | 44.02                         |
| SpaCy sm merged N=50          | 0.840                 | 70.67                         |
| SpaCy sm separate N=10        | 0.834                 | 10.12                         |
| SpaCy trf separate N=3        | 0.874                 | 6.93                          |
| SpaCy trf separate N=5        | 0.892                 | 11.68                         |
| SpaCy trf separate N=10       | 0.911                 | 23.47                         |
| Flair merged N=10             | 0.861                 | 15.54                         |
| **Flair separate N=3**        | 0.879                 | **6.27**                      |
| Flair separate N=5            | 0.895                 | 10.58                         |
| Flair separate N=10           | **0.914**             | 21.30                         |
NLI model. Comparing with existing

| Models                             | Accuracy, % | Efficiency CPU, sec per sample | Efficiency GPU, sec per sample |
|------------------------------------|-------------|--------------------------------|-------------------------------|
| SemBERT                            | 91.9        | -                              | 0.51                          |
| HBMP                               | 86.6        | -                              | 0.02                          |
| Our architecture + bert-base-uncased | 85.2        | 0.1                            | 0.006                         |
| **Our architecture + bart-base**   | **86.9**    | 0.12                           | 0.006                         |
| Our architecture + albert-base     | 84.98       | 0.08                           | 0.006                         |
| Our architecture + USE             | 78.7        | **0.036**                      | **0.004**                     |

Note: Experiments are done using CPU-only 2.0 GHz Intel instance, and RTX2070 GPU instance. Predefined splits were used.

USE - Universal sentence encoder
Trade-off between Accuracy and Speed

Efficiency of MLM models for text encoding

Accuracy on MNLI drops by ~2% when comparing large and base configurations.

Source: BERT (Devlin et al., 2018).

Note: Experiments are done using CPU-only 2.0 GHz Intel instance, 8Gb RAM
# Transfer learning approach

## Training on SNLI dataset

| Model                     | Accuracy on SNLI dataset | Accuracy on MNLI dataset |
|---------------------------|--------------------------|--------------------------|
| Siamese + bert-base-uncased | 85.20                    | 59.16                    |
| Siamese + bart-base       | **86.90**                | **63.19**                |
| Siamese + albert-base     | 84.98                    | 58.58                    |

## Training on MNLI dataset

| Model                              | Accuracy on SNLI dataset | Accuracy on MNLI dataset |
|------------------------------------|--------------------------|--------------------------|
| Siamese + bert-base-uncased        | 65.33                    | 76.10                    |
| Siamese + bart-base                | **66.93**                | **77.85**                |
| Siamese + albert-base              | 66.33                    | **80.65**                |

## Full training on specific dataset vs. training on SNLI and classifier fine tuning on FEVER and MNLI

| Model                              | MNLI classifier fine tuned vs. full train | FEVER classifier fine tuned vs. full train |
|------------------------------------|---------------------------------------------|--------------------------------------------|
| bert-base-uncased                  | 64.8% / 76.1%                              | 70.1% / 79.81%                            |
| bart-base                           | 67.6% / **77.85%**                         | **74.4%** / 85.24%                        |
| bert-base-uncased + fine tuned     | 65.4% / 76.29%                             | 69.7% / 82.45%                            |
| bart-base + fine tuned             | **68.1%** / 77.35%                         | 73.0% / **85.62%**                        |
Wikipedia domain-specific NLI model. Data preparation. Tags cleaning

Example from Wikipedia dump:

“Selena began recording professionally in 1982. Selena (film)” includes tags Selena and Selena (film).
Wikipedia domain-specific NLI model. Data preparation. Tags cleaning

Confusion matrix:

| True label                  | SUPPORTS       | REFUTES       | NOT ENOUGH INFO |
|-----------------------------|----------------|---------------|-----------------|
| SUPPORTS                    | 8553 (25.40%)  | 481 (1.43%)   | 1747 (5.19%)    |
| REFUTES                     | 2048 (6.08%)   | 6641 (19.72%) | 2444 (7.26%)    |
| NOT ENOUGH INFO             | 937 (2.78%)    | 823 (2.44%)   | 10001 (29.70%)  |

Accuracy = 0.748
Wikipedia domain-specific NLI model.

Data preparation. Filtering

**Approach:**

1. Filtered out absolute duplicates by fields 'claim' and 'hypothesis'. (8.8% reduced)
2. Balancing distribution of SUPPORTS/REFUTES classes among hypothesis sentences. (6.9% reduced)
3. Undersample NOT ENOUGH INFO class samples to the amount of major class. (12.2% reduced)

**Result:**

Distributions of labels across datasets
Wikipedia domain-specific NLI model.
Data preparation. Filtering. Results
Complete system evaluation

Original WikiCheck API flow:

Claim → NER model → Wiki API article search → Wiki API text collection → Calculating embeddings → Classification → Hypothesis & Labels

Modifications for FEVER validation:

Claim → NER model → Wiki API article search → Getting texts from Wiki dump 2017 → Calculating embeddings → Classification → Hypothesis & Labels → Aggregation logic → FEVER prediction

Note: 11.51% of articles found by MediaWiki API do not have a matched text in the dump
## Complete system evaluation

### Accuracy results:

| Team/Name          | FEVER rank | Evidence F1 | FEVER score | Accuracy |
|--------------------|------------|-------------|-------------|----------|
| UNC-NLP            | 1          | 0.5322      | 0.6398      | 0.6798   |
| UCL MRG            | 2          | 0.3521      | 0.6234      | 0.6744   |
| Athene             | 3          | 0.3733      | 0.6132      | 0.6522   |
| The Ohio St. Uni   | 7          | **0.5854**  | 0.4322      | 0.4989   |
| GESIS Cologne      | 8          | 0.1981      | 0.4058      | 0.5395   |
| WikiCheck          | -          | 0.3587      | 0.4307      | 0.5753   |

*Domain-specific NLI model*

*Improving the search*

*Building NLI model*
Complete system evaluation

Efficiency results:
Testing 1000 random claims from FEVER

Note: Experiments are done using CPU-only 2.0 GHz Intel instance
Demo
Demo

NLI model:

Input

Output
**Demo**

**Fact checking model:**

| Name     | Description          |
|----------|----------------------|
| claim    | The Earth is flat.  |

Input

```
curl -X GET "https://all.wmcloud.org/fact_checking_model/?claim=The%20Earth%20is%20flat." -H "accept: application/json"
```

Response body

```
{  
  "results": [  
    {  
      "claim": "The Earth is flat."
    }  
  ]
}
```

Output
Demo

Fact checking model + aggregation:

Input

Output
Conclusions

Main contributions

● Revealed NLI datasets limitations and annotation artifacts. Proposed the heuristic filtering technique that led to the model's accuracy increase.

● Showed that usage of NER models for search increases the quality of results.

● Proposed accurate and efficient sentence-based NLI model.

● Discovered that full model training on specific dataset is required to get the best results. Proposed unsupervised fine-tuning of MLM for domain adaptation.
Conclusions

Successfully reached the main goal of the thesis:

- Transformed academic research into a practical tool.
- Presented WikiCheck API
- Made all the code for WikiCheck API available on the Github.

WikiCheck API: https://nli.wmcloud.org
WikiCheck github: https://github.com/trokhymovych/WikiCheck
Future work

- Experiment with NER models, types of entities used for query enhancing. Consider (POS) tagger usage for keywords extraction.
- Experiment with different methods of sentence embeddings creation.
- Experiment with more complex classifier models (last layer of the NLI model) and larger MLM encoders.
- Observe the relation between the length of the hypothesis and the NLI model accuracy.
- Aggregation phase modifications research.
- Tune the efficiency of embeddings calculation by MLM size reduction, model distillation, float parameters quantization.
Questions?
Thank you for attention

Contact:

WikiCheck API:

Gold Coast, Queensland, Australia, 1-5 November 2021