Statistical Claim Checking: StatCheck in Action (demonstration)

Oana Balalau, Simon Ebel, Théo Galizzi, Ioana Manolescu, Quentin Massonnat, Antoine Deiana, Emilie Gautreau, Antoine Krempf, Thomas Pontillon, Gérald Roux, et al.

HAL Id: hal-03767992
https://hal.inria.fr/hal-03767992

Submitted on 2 Sep 2022

Statistical Claim Checking: StatCheck in Action

Oana Balalau, Simon Ebel, Théo Galizzi,
Ioana Manolescu, Quentin Massonnat
firstname.lastname@inria.fr
Inria and Institut Polytechnique de Paris
France

Antoine Deiana, Emilie Gautreau, Antoine
Krempf, Thomas Pointillon, Gérald Roux, Joanna
Yakin
firstname.lastname@radiofrance.com
France Info, Radio France
France

ABSTRACT

To strengthen public trust and counter disinformation, *computational fact-checking*, leveraging digital data sources, attracts interest from the journalists and the computer science community. A particular class of interesting data sources is *statistics*, that is, numerical data compiled mostly by governments, administrations, and international organizations. Statistics typically are multidimensional datasets, where multiple dimensions characterize one value, and the dimensions may be organized in a hierarchy.

We developed StatCheck, a fact-checking system specialized in French. The technical novelty of StatCheck is twofold: (*i*) we focus on multidimensional, complex-structure statistics, which have received little attention so far, despite their practical importance; and (*ii*) novel statistical claim extraction modules for French, an area where few resources exist. We will demonstrate our system on large statistic datasets (hundreds of millions of facts), including the complete INSEE (French) and Eurostat (European Union) datasets.

A video demonstration of our system is available online[1].

1 INTRODUCTION

Professional journalism work has always involved verifying information with the help of trusted sources. In recent years, the proliferation of media in which public figures make statements, in particular online, has lead to an explosion in the amount of content that may need to be verified to distinguish accurate from inaccurate, and even potentially dangerous, information.

To help journalists deal with the deluge of information, computational fact-checking [CLL+18, NCH+21] emerges as a growing, multidisciplinary field. The main tasks of a fact-checking system are: *identifying the claims* made in an input document, *finding the relevant evidence* from a reference corpus, and (optionally) *producing an automated verdict* (is the claim true or false?). A reference corpus can be a knowledge graph [CSR+15], Web sources such as Wikipedia [NCB19, YMW+18], or relational tables [CWC+20, HNM+20, JTY+19, KSPT20].

For fact-checks to be convincing, professional journalists prefer reference sources of high quality, carefully built by specialists. These include statistics produced by governmental and international organizations, such as INSEE (the French national statistics institute) and Eurostat (the equivalent EU office). Technically speaking, such statistics are *multidimensional tables*, where a *fact* is a number, characterized by one or more a *dimensions*, such as a geographical

unit, time interval, and other categories such as "Education level". Such data sources are significantly more complex than relational tables, making their usage challenging. Consequently, despite the interest in such sources, few works have used them for automatic fact-checking [CMT18, DCMT19].

We propose to demonstrate StatCheck, a fact-checking system specialized in the French media. We developed StatCheck in a collaboration between computer science researchers and journalists at Radio France, the French national radio, with a daily audience recently estimated at 15.8M users[2]. StatCheck builds upon the open-source code base of [CMT18, DCMT19], which it generalizes into a first **generic fact-checking pipeline based on multidimensional statistics**. Different from [CWC+20, HNM+20, JTY+19, KSPT20, CSR+15, NCB19, YMW+18], StatCheck also includes a claim detection step, which saves journalists' time by focusing their attention on the claims worth checking; our claim detection module significantly outperforms the only one we know of for French [DCMT19]. The fact-checking journalist authors prefer to interpret the relationship between the known statistics and the statistic claim StatCheck identifies and include these interpretations in various news segments they author.

Outline Below, we present the actual organization of statistic databases, and the StatCheck architecture, in Section 2. Then, we explain how this architecture is instantiated over two different sources, INSEE and Eurostat, whose size and organization significantly varies, in Section 3; we ingest and index all the data to support efficient search over it, as described in Section 4. We find claim by subscribing to media sources, or allowing users to upload their own content (Section 5). We then describe the proposed demonstration scenarios (Section 6).

2 FACT-CHECKING BASED ON MULTIDIMENSIONAL STATISTICS

A multidimensional dataset consists of a set of *facts*, each having one *value* along a set of *dimensions*. For instance, Figure 1 (top) represents three-dimensional datasets: French departments are on the horizontal axis, education levels on the vertical axis, while years are on the third (depth) axis. In each cell, the dataset could store the number of students in the respective department, level of study, and year. However, the actual Open Data statistics published by the government or international organizations are typically much more complex, as shown at the bottom of Figure 1. First, to save space, *dimension values may be encoded* into short codes, e.g., "HI" for "High school", "MI" for "Middle school"; a decoding dictionary, associating a human-understandable term to each code, is published

---

[1]https://drive.google.com/drive/folders/16b202dFIVR3uSqmsnDQSQ7FkOsgJdZ6e?usp=sharing

[2]https://www.radiofrance.com/professionnels/regie-publicitaire/actualite/les-audiences-janvier-mars-2022
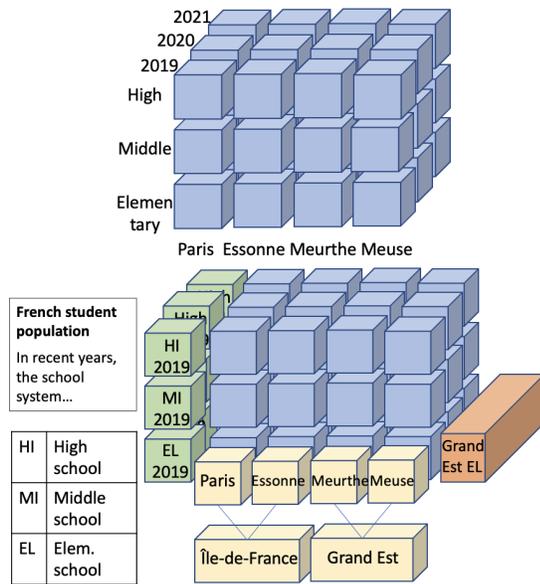
Figure 1: Multidimensional statistic data: conceptual view (top), structure of actual published dataset (bottom).

with, or close to the data cells. Although not shown in the figure, *dimension names* are similarly encoded. Second, *header cells*, shown in yellow and green in the figure, *may be mixed with data cells*; this requires effort to interpret them correctly. Note also that *there can be a hierarchy of headers*, e.g., a dataset at the granularity of departments may also include region names, e.g., "Île-de-France" and "Grand Est", placed in the data files above, or close to, the region header cells. Third, datasets may contain *partially aggregated results* next to the cell-level data, illustrated by the orange box holding the sum of all facts for one region (Grand Est), one education level (elementary), and the three years. Fourth, for each dataset, there may exist a separate, textual *description*, which contains a title, e.g., "French student population", and other comments.

Data representation in files. In practice, a multidimensional statistic dataset is published as a file, which can be CSV, a spreadsheet, etc. The dataset is laid out in a bidimensional format, with facts on each line and as many lines as needed. If the data has more than two dimensions, which is often the case, this leads to *row header cells encoding several dimensions and their values*, such as "HI 2019", "MI 2019" etc., in the figure. The file may start with the column headers (yellow), then the encoded multidimensional row header cell "EL 2019" followed by the four cells corresponding to it, then a similar line for "MI 2019", a line for "HI 2019", followed by similar lines for 2020, then 2021, etc. Partially aggregated results are interspersed between such lines.

Challenges and architecture. To exploit such datasets for fact-checking, a set of challenges must be addressed. The useful information, e.g., "How many elementary school students were in the Île-de-France region in 2019?", is a number in a cell. To find such information, we must **identify and store its relationships with human-understandable descriptions of its dimensions**, such as "Education level: Elementary school". In this example, the question is asked at a granularity (region) more coarse than the granularity of the data. To find the answer, we must exploit the fact

that Paris and Essonne are departments in the Île-de-France region. Further, statistic claims may use similar but different language, e.g., a claim may be made about "pupils in Île-de-France". **Linguistic knowledge must be leveraged** to connect the claim terminology with that of the dataset. **Fine-granularity answers are preferred** that is: if the answer consists of one or a few cells only, those should be returned to avoid users' efforts to search through facts in a file (a dataset can have millions of lines). Finally, **speed at scale is essential** to enable journalists to work efficiently.

To address these challenges, we have devised an architecture shown in Figure 2. The modules in the lower row acquire reference datasets and analyze and index them. Those in the upper row acquire content to be fact-checked, extract claims, and identify,the most pertinent stored facts to use to check the claims.

3 STATISTIC FACT DATABASE

We crawled the INSEE [www22b] and Eurostat [www22a] Web sites, extracting and storing their complete statistics as follows.

INSEE publishes each statistic report as an HTML page containing a description (title and comments) and statistic tables, in Excel or HTML. As of May 2022, there are 60,002 Excel files (each of which may contain several tables) and 58,849 HTML tables. *The table organization varies significantly across the datasets*; nested headers are frequent. The largest table has 50.885 lines. Following [CMT17], to capture all the elements of an INSEE dataset, we turn it into an **RDF graph**, where each data cell, header cell, and partial aggregate becomes an RDF node (URI). Further, each data cell or partial aggregate node is connected, through an RDF triple, to the cells corresponding to its closest header cells. Thus, the number of elementary school students in Paris in 2019 is connected to header cells labeled "Paris", respectively, "Elementary school 2019" - observe that we decoded "EL 2019" using the dictionary. Further, each header cell is connected through an RDF triple to its parent header cell. This allows us to quickly find out that the elementary school students in Paris in 2019 are also counted as being in the Île-de-France region. Finally, we create an RDF node per dataset, connected to all its header cells and the textual title and comments (each modeled as an RDF literal). The INSEE corpus lead to **7,362,538,629 RDF triples**, including **22,366,376 header cells**. We store them in Apache's Fuseki (with TDB2) RDF store.

Eurostat publishes 6,803 statistic tables, ranging from 2 lines to 37 million lines, and 580 dictionaries that, together, decode 243,083 statistical concepts codes into natural-language descriptions. Together, the data files total **414.908.786 lines**. In Eurostat, *dimension hierarchies are described in the dictionaries*; we store them in memory. The statistic tables are simple TSV files; we keep them in this format, complemented by specialized indexes, as we explain below.

4 STATISTIC SEARCH

Given a keyword query $Q = \{k_1, k_2, \ldots, k_n\}$, such as "middle school pupils in Île-de-France in 2020", we find: the most relevant facts from our INSEE and Eurostat corpus; or, if a specific fact is not found, but some datasets appear related to the query, return those datasets. In general, there may be both fact-level and dataset-level answers; we return a ranked list based on their relevance.

We call **metadata of a statistic dataset** all the natural-language elements part of or associated with the dataset: its title, comments, and human-understandable versions of all its header values. We
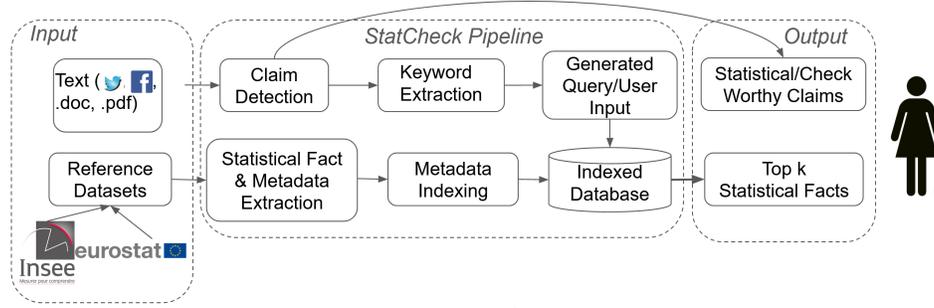
Figure 2: STATCHECK architecture overview.

use $\mathcal{L} = \{T, H, C\}$ to denote the set of the **locations** in which a term can appear in metadata, respectively: the dataset title, a header, or a comment. The locations are important: (*i*) since a term appearing in a title is more important than one appearing in a header (Section 4.1); (*ii*) to determine if a dataset matches some keywords headers of different dimensions - in which case the cell at the intersection of those dimensions likely has a very pertinent result (Section 4.2).

4.1 Dataset Indexing and Search

We split the metadata of each dataset d into a set of tokens $T = \{t_1, \ldots, t_D\}$. For each token t, we identify, based on a Word2Vec model, the 50 tokens t' closest semantically to t.

Next, for each appearance of a token t in location l within an **INSEE** dataset d, our **term-location index** I_{TL} stores: an entry of the form (t, d, l), and 50 entries of the form $(t', d, l, dist)$, for the 50 tokens closest to t. For instance, when t is "*school*", t' could be "*teacher*", "*pupil*", "*student*", etc. For fast access, I_{TL} is stored in the Redis in-memory key-value store.

The large size of **Eurostat** statistics prevents cell- or row-level metadata indexing, as the index might outgrow the memory. So instead, we index *occurrences of statistical concept codes in datasets*, as follows. Let c be a Eurostat concept, e.g., "EL", appearing in dataset d at a location $l \in \mathcal{L}$, and d_c be the decoding of c, e.g., "*Elementary school*" for "EL". Let $T_{dc} = \{t_1, t_2, \ldots, t_N\}$ be the tokens in d_c, and for $1 \leq i \leq N$, let t_i^j, for $1 \leq j \leq 50$, be the tokens closest to t_i. For each $t_i \in T_{dc}$, we insert in the **term-dataset index** I_T, also stored in Redis: a (t_i, d, l) entry; and, for every t_i^j similar to t_i, an entry $(t_i^j, d, l, dist, t_i)$, where $dist$ is the distance between t_i and t_i^j.

Given the query $Q = \{k_1, \ldots, k_n\}$, we search I_{CL} and I_T for entries of the form (k_i, d, l) or $(k_i, d, l, dist, k_i')$ for each k_i. We rank datasets based on the relevance score introduced in [CMT18]. It leverages the word distances between the query keywords and the datasets' metadata and also reflects the locations where the keywords were found for each dataset. Among the retrieved datasets, we keep the 20 having the highest score.

4.2 Data Cell Indexing and Search

Our next task is to extract results at the finest granularity level possible. Let d be one of the most interesting datasets, and $I(d)$ be the set of **all index entries for the query Q and d**. For our sample query Q and dataset in Figure 1, $I(d)$ contains:

- For "middle school", header (H) entries for "Middle school" (exact), as well as for "High school" and "Elementary school"

(similar); a title (T) entry for "student" (similar); and a comment (C) entry for "school" (similar);
- For "pupils", H, T, and C entries for the similar words above;
- For "Île-de-France", an exact H entry, and two similar H entries for "Paris" and "Essonne";
- For "2020", exact H entries.

If $I(d)$ only features title (T) or comment (C) locations, then d is pertinent as a whole, and no cell search is needed.

On the contrary, if $I(d)$ has several header entries (having $l = H$), matching two or more distinct query keywords (or close terms), this means that d holds some fine-granularity results for the query. If $I(d)$ holds an entry along each dataset dimension d, they designate exactly one cell, and we can directly return its value. Otherwise, the result is a collection of all the cells from d characterized by the dimension values designated by the entries in $I(d)$. In our example, we should return the cells for "MI 2019", "2020", and locations "Paris" and "Essonne", which belong to Île-de-France.

- If d is an INSEE dataset, $I(d)$ specifies exactly which rows and columns are concerned. Then, the cell is identified by asking a SPARQL query [CMT18], evaluated by Fuseki.
- On the contrary, if d is an Eurostat dataset, $I(d)$ only specifies that "*some* row (column) headers match". Identifying the relevant cells requires more effort, as we explain below.

An Eurostat file has at most a few dozen columns. To find the *column* referred to by an $I(d)$ entry whose key is k, we search for k in the first (header) line of d. To efficiently find the *row* even in a huge file, we created another index on all the Eurostat data files, inspired by the Adaptive Positional Map of [ABB+15], storing the positions of the data rows in d containing k in their header. Knowing the rows and column indexes, we read the relevant row(s) from d, and extract from them the relevant data cell(s). On both INSEE and Eurostat, this takes from a few microseconds to 2.5s.

5 CLAIM DETECTION

A claim is a statement to be validated. The validation is achieved by finding related statements, called evidence, which back up or disprove the claim. In our work, the claims are detected in an input text, while the evidence is retrieved from a set of trusted sources, our reference datasets.

5.1 Statistical Claim Detection

In [DCMT19], the authors introduced a statistical claim detection method that given an input set of statistical entities (e.g. *chômage*, *coefficient budgétaire*) and a sentence, it retrieves all the *statistical statements* of the form ⟨statistical entity, numerical value and

Figure 3: Screen captures of StatCheck' GUI. Top: statistic search interface with sample query results: data cells with row header in blue and column header in red; bottom: tweet analysis interface.

unit, date⟩ present in the sentence. *The statistical statement, if present, represents the statistical claim to be verified.* The statistical entities and units are retrieved using exact string matching, while the date is extracted using HeidelTime [SG10], a time expression parser. If the parser finds no date, the posting timestamp is used. The initial statistical entity list is constructed from the reference datasets by taking groups of tokens from the headers of tables, we refer to [DCMT19] for more details.

We improved the method presented in [DCMT19] to optimize both the speed and quality of extractions. We refer to the two methods as OriginalStatClaim [DCMT19] and StatClaim. We first performed a more careful match between the tokens of a sentence and our input statistical entities. Then, using the syntactic tree of the sentence and a lemmatizer, statistical entities are matched using their lemma, and are extended to contain the entire nominal group of the matched token. Numerical values are associated with units using both lemmas matching from our set of units and syntactic analysis. As in the original approach, if we retrieve a statistical statement of the form ⟨statistical entity, numerical value, and unit, date⟩, we have found a claim to verify. In the default setting of our algorithm, a claim should contain all three elements.

5.2 Check-worthy Claim Detection

To complement the statistical claim detection model, we developed a model that is not conditioned on a set of initial statistical entities. Instead, the model classifies a sentence as check-worthy or not, where check-worthiness is defined as *sentences containing factual claims that the general public will be interested in learning about their veracity* [AHLT20]. We leveraged the ClaimBuster dataset [AHLT20], containing check-worthy claims in English from the U.S. Presidential debates, to train a cross-lingual language model, XLM-R [CKG+19], which can perform zero-shot classification on French sentences after having been trained on English data.

The ClaimBuster dataset. ClaimBuster is a crowd-sourced dataset of 11K sentences from the 15 U.S. presidential elections debates from 1960 to 2016 that have been annotated. Each sentence is labeled as check-worthy or not; we use them to fine-tune the XLM-R model.
Classification. The XLM-R model is a Transformer-based language model which achieves state-of-the-art results on multilingual tasks

Dataset	Precision	Recall	F1 score
ClaimBuster	0.883	0.848	0.865
French tweets	0.612	0.769	0.682

Table 1: Evaluation of the fine-tuned XLM-R model.

such as the XNLI benchmark [CRL+18], while remaining competitive on monolingual tasks. We used a weighted cross-entropy loss to account for the unbalanced ratio of labels. The dataset was split into train, dev, and test datasets with a ratio of 80%/%10%/10%. We fine-tune the model using a learning rate of $5 \cdot 10^{-5}$, a batch size of 64, and the AdamW optimizer.
Evaluation. To evaluate the performance of the different models on French data, we randomly sampled 200 French tweets and labeled them as check-worthy or not following the definition in [AHLT20]. The Cohen Kappa score for inter-annotator agreement is 0.6, signifying moderate to substantial agreement. The results can be found in Table 1. The drop in precision on French data could be because we are evaluating on a small test dataset or because the tweets' format and vocabulary might differ from the ones in the training dataset.

6 DEMONSTRATION SCENARIOS

Our system is developed in Python and deployed on a Unix server. Its GUI is accessible via a Web server; Figure 3 illustrates it.

Demonstration attendees will be able to: (*i*) Ask queries in the statistic search interface, and inspect the results, at the level of cell, line, or column, together with their metadata from the original statistic site (INSEE or Eurostat); (*ii*) Visualize the analysis of incoming social media messages (as they arrive in real-time), in order to see the statistical mentions and claims deemed potentially check-worthy, identified in these messages; StatCheck also proposes candidate queries for the statistic search interface, as shown in Figure 3. (*iii*) Select various options (restrict to numerical claims or not, include statements about the future or not, include first-person texts or not, etc.) and see their impact on the claim extraction output. (*iv*) Write their own text and/or suggest other content to be processed by our analysis pipeline (Section 5).

For the non-French speaking audience, we will use Google Translate for social media messages, and prepare examples where French and English statistic terms are sufficiently close, e.g., *industrie*, etc.

REFERENCES

[ABB+15] Ioannis Alagiannis, Renata Borovica-Gajic, Miguel Branco, Stratos Idreos, and Anastasia Ailamaki. Nodb: efficient query execution on raw data files. *Commun. ACM*, 58(12):112–121, 2015.

[AHLT20] Fatma Arslan, Naeemul Hassan, Chengkai Li, and Mark Tremayne. A Benchmark Dataset of Check-worthy Factual Claims. In *14th International AAAI Conference on Web and Social Media*. AAAI, 2020.

[CKG+19] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.

[CLL+18] Sylvie Cazalens, Philippe Lamarre, Julien Leblay, Ioana Manolescu, and Xavier Tannier. A content management perspective on fact-checking. In *WWW (Companion Volume)*, pages 565–574. ACM, 2018.

[CMT17] Tien Duc Cao, Ioana Manolescu, and Xavier Tannier. Extracting linked data from statistic spreadsheets. In *International Workshop on Semantic Big Data*, International Workshop on Semantic Big Data, pages 1 – 5, 2017.

[CMT18] Tien-Duc Cao, Ioana Manolescu, and Xavier Tannier. Searching for Truth in a Database of Statistics. In *WebDB*, pages 1–6, 2018. Code available at: https://gitlab.inria.fr/cedar/excel-search.

[CRL+18] Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. Xnli: Evaluating cross-lingual sentence representations. In *EMNLP*, 2018.

[CSR+15] Gianluca Ciampaglia, Prashant Shiralkar, Luis M. Rocha, Johan Bollen, Filippo Menczer Menczer, and Alessandro Flammini. Computational fact checking from knowledge networks. *PLoS ONE*, 10(6), 2015.

[CWC+20] Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. Tabfact : A large-scale dataset for table-based fact verification. In *ICLR*, 2020.

[DCMT19] Tien Duc Cao, Ioana Manolescu, and Xavier Tannier. Extracting statistical mentions from textual claims to provide trusted content. In *NLDB*, 2019. Code available at: https://gitlab.inria.fr/cedar/statstical_mentions.

[HNM+20] Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. TaPas: Weakly supervised table parsing via pre-training. In *ACL*, pages 4320–4333, 2020.

[JTY+19] Saehan Jo, Immanuel Trummer, Weicheng Yu, Xuezhi Wang, Cong Yu, Daniel Liu, and Niyati Mehta. Verifying text summaries of relational data sets. In *SIGMOD*, SIGMOD '19, page 299–316, 2019.

[KSPT20] Georgios Karagiannis, Mohammed Saeed, Paolo Papotti, and Immanuel Trummer. Scrutinizer: Fact checking statistical claims. *Proc. VLDB Endow.*, 13(12):2965–2968, 2020.

[NCB19] Yixin Nie, Haonan Chen, and Mohit Bansal. Combining fact extraction and verification with neural semantic matching networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence*, pages 6859–6866. AAAI Press, 2019.

[NCH+21] Preslav Nakov, David Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barrón-Cedeño, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. Automated fact-checking for assisting human fact-checkers. In *IJCAI*, pages 4551–4558, 2021. Survey Track.

[SG10] Jannik Strötgen and Michael Gertz. Heideltime: High quality rule-based extraction and normalization of temporal expressions. In *Int'l. Workshop on Semantic Evaluation*, pages 321–324, 2010.

[www22a] Statistic office of the european commission. https://https://ec.europa.eu/eurostat, 2022.

[www22b] Institut national de statistiques et études économiques. https://www.insee.fr, 2022.

[YMW+18] Takuma Yoneda, Jeff Mitchell, Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. UCL machine reading group: Four factor framework for fact finding (HexaF). In *FEVER*, pages 97–102, Brussels, Belgium, 2018. Association for Computational Linguistics.