Fact-checking Multidimensional Statistic Claims in French

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

Fact-checking Multidimensional Statistic Claims in French

Oana Balalau, Simon Ebel, Théo Galizzi,
Ioana Manolescu, Quentin Massonnat
Inria and Institut Polytechnique de Paris
1 rue Estienne Honoré d'Orves,
91120 Palaiseau, France
firstname.lastname@inria.fr

Antoine Deiana, Emilie Gautreau, Antoine Krempf,
Thomas Pontillon, Gérald Roux, Joanna Yakin,
FranceInfo, Radio France
116 Av. du Président Kennedy,
75016 Paris, France
firstname.lastname@radiofrance.com

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 comprises *statistics*, that is, numerical data compiled mostly by governments, administrations, and international organizations. Statistics are often *multidimensional datasets*, where multiple dimensions characterize one value, and the dimensions may be organized in hierarchies.

This paper describes StatCheck, a statistic fact-checking system jointly developed by the authors, which are either computer science researchers or fact-checking journalists working for a French-language media with a daily audience of more than 15 millions (aud, 2022). 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 validate the efficiency and quality of our system on large statistic datasets (hundreds of millions of facts), including the complete INSEE (French) and Eurostat (European Union) datasets, as well as French presidential election debates.

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 led 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 (Cazalens et al., 2018; Nakov et al., 2021) 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 (Ciampaglia et al., 2015), Web sources such as Wikipedia (Nie et al., 2019; Yoneda et al., 2018), or relational tables (Chen et al., 2020; Herzig et al., 2020; Jo et al., 2019; Karagiannis et al., 2020).

For fact-checks to be convincing, professional journalists prefer reference sources of high quality, carefully built by specialists. These include **statistics** produced and shared by governmental and international organizations, such as INSEE, the French national statistics institute [1] and Eurostat, the equivalent European Union office [2]. 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", etc. Unfortunately, such data sources are significantly more complex than relational tables, making their usage challenging. Consequently, despite the interest in such sources, only a few works have used them for automatic fact-checking (Cao et al., 2018; Duc Cao et al., 2019).

In our collaboration between computer scientists and fact-checking journalists, we have developed, deployed, and continue to be extending StatCheck, a fact-checking system specialized in the French media arena. StatCheck builds

[1] https://www.insee.fr
[2] https://https://ec.europa.eu/eurostat

upon the open-source code base of (Cao et al., 2018; Duc Cao et al., 2019). We significantly improved its data ingestion speed and more than doubled its statistic corpus by adding Eurostat data. Different from (Chen et al., 2020; Herzig et al., 2020; Jo et al., 2019; Karagiannis et al., 2020; Ciampaglia et al., 2015; Nie et al., 2019; Yoneda et al., 2018; Aly et al., 2021), 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 (Duc Cao et al., 2019).

Outline. Below, we start by presenting a set of functional requirements derived from the journalist authors' experience in Section 2. Next, we describe the actual organization of statistic databases, and the STATCHECK architecture, in Section 3. Then, we explain how this architecture is instantiated over two different sources, INSEE and Eurostat, whose size and organization significantly vary, in Section 4; we ingest and index all the data to support efficient search over it (Section 5). Finally, our claim detection modules are described in Section 6, then we conclude.

2 Fact-Checking Work Routine and Requirements

The journalist authors are part of the same team, specializing in fact-finding and fact-checking in a French-speaking national media. The material they author is disseminated through both the native and online media channels of their news organization. Their work is split among the two main classes identified in (Juneja and Mitra, 2022): *short-term claim centric*, focusing on the veracity of statements made continuously by public figures, which need to be checked relatively quickly; and *long-term issue-centric*, whereas individual journalists maintain and increase their knowledge of application topics, such as "law enforcement", "education and research", "defense", etc.

The short-term, claim-centric work raises several requirements. First, journalists know whose claims might interest their audience. Thus, they need an *interesting subset (selection) of social media content* to be made available through a Web platform. Journalists specify a set of social media account handles (currently Twitter and Facebook), and need the ability to modify this set themselves, as people gravitate in and out of the public's attention. Second, whenever claims about statistic entities are made in this social media content sphere, *bringing these claims to their attention*, isolating them from the mass of social communication of the figures they follow, saves journalists time and effort. Third, as previously noted in (Cazalens et al., 2018; Saeed and Papotti, 2021), data sources relevant to a given claim must be quickly identified and as precisely as possible. This again saves journalists time to search statistic data sources that may be very large, i.e., Eurostat publishes thousands of datasets, some with millions of rows.

The long-term, issue-centric work also benefits from these functionalities, yet it is more open; journalists may peruse claims for which they have not identified relevant sources yet, but still *appreciate a recommendation of most likely check-worthy claims*. User-friendly means to filter messages considered check-worthy (Should messages about the future, such as electoral promises, be considered, or not? Is a number required in a statistical claim, or not?) are also appreciated.

Common to both kinds of work, the newsroom involved in this project has the core tenet that *any verdict or judgment must be vetted by journalists*, since publishing it engages their professional responsibility. This has a set of consequences. (i) Journalists need to analyze the facts relevant to a claim and interpret them in a nuanced way for their audience. For instance, a difference of 5% between a number stated in a claim, and the value in a reference source, may be negligible or, on the contrary, a serious attempt to mislead, depending on the context. Thus, unlike prior systems (Chen et al., 2020; Herzig et al., 2020; Jo et al., 2019; Karagiannis et al., 2020), STATCHECK does not compute a "true/false" verdict, leaving this tasks to journalists. (ii) For transparency and trust, links to any fact on its original publishing site must be provided together in the fact-check.

3 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 a three-dimensional dataset: 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 stores the number of students in the respective department, level of

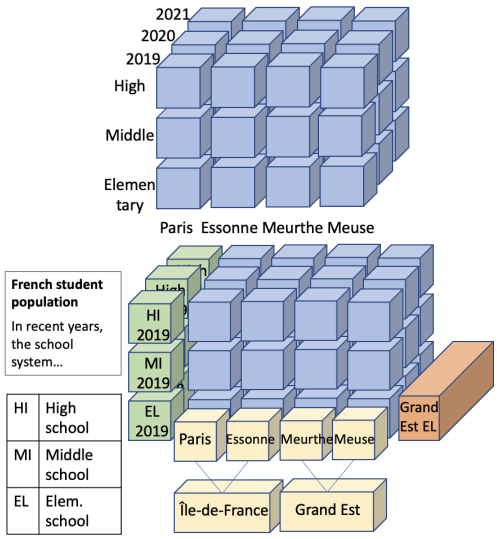


Figure 1: Multidimensional statistic data: conceptual view (top), structure of actual published dataset (bottom).

study, and year. In practice, 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", etc.; a decoding dictionary, associating a human-understandable term to each code, is published 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 also contain *partially aggregated results*, 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. For that, it is laid out in a bidimensional format, with some 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. 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 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) that is 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. As mentioned in our requirements (Section 2), **fine-granularity answers are preferred**, that is: if the answer consists of one or a few cells only, those should be extracted from the dataset and returned, to avoid journalists' efforts to search in potentially large files. Finally, **speed at scale is important**, to enable journalists to work efficiently.

Architecture. To address these challenges, based on the requirements described in Section 2, we have devised an architecture shown in Figure 2. The modules in the lower row acquire and process reference datasets (Section 4), e.g., statistics about education in France. Those in the upper row acquire content to be fact-checked, e.g., a tweet stating: "*More teachers are needed to educate 200K pupils in Île-de-France!*", extract claims (Section 6), in this case "*200K pupils in Île-de-France*", and identify the most relevant facts for checking these claims, by searching the appropriately indexed reference datasets (Section 5).

4 Statistic Fact Database and Storage

By crawling, we acquired the complete INSEE and Eurostat statistics, and store them as follows.

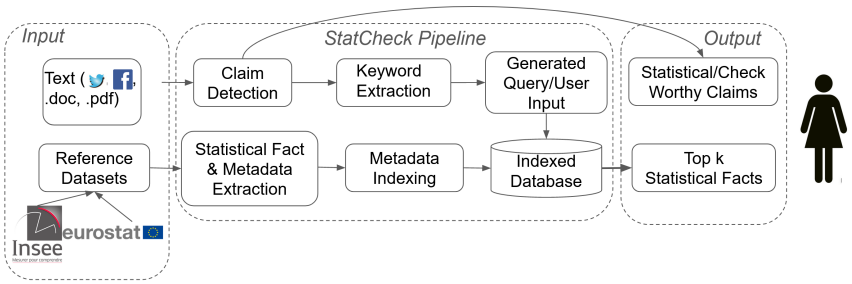


Figure 2: StatCheck architecture overview.

INSEE publishes each statistic report as an HTML page containing a description (title and comments on the data), and statistic tables in Excel or in 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 (Cao et al., 2017), to capture all the elements of an INSEE dataset, we turn it in an **RDF graph** (www-rdf), where each data cell, header cell, and partial aggregate becomes an RDF node (URI). 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" (where "EL 2019" was decoded using the dictionary). Finally, each header cell is connected through an RDF triple to its parent header cell. This allows us to easily find out that the elementary school students in Paris in 2019 are also to be counted as being in the Île-de-France region. We also create an RDF node per dataset, which is connected to all its header cells and to 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 the Jena Fuseki server with the TDB2 persistent back-end (www-tdb2).

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, all of which we acquired in StatCheck' database. Together, the Eurostat data files total **414.908.786 lines**. In Eurostat, *dimension hierarchies are described in the dictionaries*; we store these in memory. The statistic tables are simple-structure TSV files, thus, storing each of them as a table in a relational database was an option. However, their number is relatively high, and storing a file in a database inevitably increases its storage footprint. Therefore, to keep the data more compact, in view also of future extensions of our platform with more statistics from the World Health Organization, World Development Index etc., we store them as plain files, complemented by specialized indexes, as we explain below.

5 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", the task we consider here is to find:

- the most relevant facts from our complete INSEE and Eurostat corpus;
- or, if a concrete fact is not found, but some datasets as a whole appear related to the query, return those datasets.

There may be several fact- (or cell-) level as well as dataset-level answers; we return a ranked list based on their relevance.

We call **metadata of a statistic dataset** all the natural-language elements that are part of or associated with the dataset: its title, comments, and human-understandable versions of all its header values. We use $\mathcal{L} = \{T, C, H\}$ to denote the set of the **locations** in which a term can appear in metadata, respectively: the dataset title, a comment, or a header. The locations are important since a term appearing in a title is more significant than one appearing in a header, and we exploit this when retrieving the datasets most relevant for a query (Section 5.1). Also, locations help determine whether a dataset matches some keywords headers of different dimensions - in which case the cell(s) at the intersection of those dimensions likely have a very relevant result (Section 5.2).

5.1 Dataset Indexing and Search

We split the metadata of each dataset d into a set of tokens $T = \{t_1, \ldots, t_N\}$, and remove stop words. 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 a location l within d, our **term-location index** I_{TL} stores: (i) the index entry (t, d, l) corresponding to the token actually found in d; and (ii) 50 entries of the form $(t', d, l, dist)$, for the 50 tokens closest to t. These extra entries enable answering queries on terms close to (but disjoint from) the dataset content. For instance, when t is "*school*", t' could be "*teacher*", "*pupil*", "*student*", etc. For fast access, I_{TL} is hosted in the Redis in-memory key-value store (www-redis). To find the datasets relevant for the query Q, we look up the query keywords in I_{TL}, and consider relevant any dataset associated with at least one keyword.

The above indexing mechanism leverages word distances. Separately, we used *geographic resources*, in particular (Eurostat, 2022) for EU locations, to make our system aware of the relationships between geographic units (cities, departments, regions) across Europe. This ensures that a dataset is considered relevant if it mentions a geographic unit that includes or is included in the query. It is important to identify geographic names in the metadata. We have adopted the FlashText algorithm (Singh, 2017), capable of finding, in a dataset metadata of size N, one of M fixed keywords in $O(N)$ time complexity. This is much faster than the $O(NM)$ cost of regular expression pattern matching used in the previous system (Cao et al., 2018) and significantly sped up indexing of the INSEE corpus[3].

Coarser-grain indexing of Eurostat statistics The large size of this corpus prevents cell- or row-level metadata indexing, as the index might outgrow the memory. 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;
- 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.

Indexing the complete Eurostat data in this way took around 4 minutes.

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 $(k_i, d, l, dist, k_i')$. Any dataset having an entry for at least one k_i is potentially interesting; we retain the 20 highest-score ones.

Dataset ranking We rank datasets based on the relevance score introduced in (Cao et al., 2018). It is a weighted combination of the word distances between the query keywords and the datasets' metadata; the weights reflect the locations where relevant terms appear in each dataset. We have also experimented with the classic BM25 (Robertson and Zaragoza, 2009) computed over all the datasets' metadata, but the results were less good, in particular, because BM25 does not handle synonyms well. We also considered embedding the query and the metadata using Sentence-Bert (Reimers and Gurevych, 2019) and comparing these with the query embedding, but opted not to use it because, for our purposes, the term location in the metadata is important, and treating the metadata as a single text loses this information.

5.2 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.

[3] Together with other optimizations related to batching calls to the Spacy tokenizer and pipelining the indexing with the data acquisition process, this brought the total INSEE indexing time from 29 hours to 4 minutes.

If $I(d)$ only features title (T) or comment (C) locations, then d is pertinent as a whole, and we do not search for cell-level answers.

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, these entries, together, designate exactly one cell, which we should return. 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. For that:

1. If d is an INSEE dataset, $I(d)$ contains the headers of the respective row and column headers. Then, the cell is identified by asking a SPARQL (W3C, 2013) query, evaluated by Fuseki, as in (Cao et al., 2018). The query requests "all the data cells from dataset d whose closest header cells are those from $I(d)$".

2. If d is an Eurostat dataset, $I(d)$ only specifies that "*some* row (column) headers match", and more effort is needed to identify the relevant cells. A Eurostat file has at most a few dozen columns, but it may have tens of millions of rows.

    - 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 identify the relevant *rows* efficiently, we created another index I_R on the Eurostat data files, inspired by the Adaptive Positional Map of (Alagiannis et al., 2015). I_R stores the positions, in the data file of d, of the rows containing a certain keyword k in their header. We store I_R directly as a binary file on disk.
    - Knowing the rows and column indexes, we read those row(s) from d, and extract from them the relevant data cell(s).

Using Fuseki, cell extraction takes 35ms up to 2.86s. On Eurostat, using I_R, we record 4.76μs up to 2.66s. The lower bound is higher for INSEE because we have to pass SPARQL queries across a connection to the Fuseki server.

6 Claim Detection

A claim is a statement to be validated, that is, we aim to establish if it is true or false. 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. Our platform detects claims from text stored in *.txt*, *.odt*, *.docx* or *.pdf* files, and from the Twitter and Facebook posts of public figures. Our platform regularly retrieves the most recent updates of a predefined group of users for posts.

6.1 Statistical Claim Detection

Previous work addresses statistical claim detection in a supervised manner by predicting statistical entity-value pair from text patterns (Vlachos and Riedel, 2015). In (Duc Cao et al., 2019), 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 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 (Strötgen and Gertz, 2010), a time expression parser. If the parser finds no date, the posting timestamp is used. More context about the claim to be verified is found using a Named Entity Recognition (NER) model, which returns organizations and locations. We note, however, that the organization and location are optional, while a statistical statement is not complete without one of its three elements. The initial statistical entity list is constructed from the reference datasets by taking groups of tokens from the headers of tables, we refer to (Duc Cao et al., 2019) for more details.

We improved this method to optimize both its speed and the quality of extractions. We refer to the two methods as OriginalStatClaim (Duc Cao et al., 2019) and StatClaim. We first performed a more careful match between the tokens of a sentence and our input statistical entities. 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 val-

ues are associated with units using both lemmas matching from our set of units and syntactic analysis. Units can be a noun following a numerical value or a nominal group containing one or more units. (e.g. "*millions d'euros*"). 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. In addition, we filter out claims from sentences whose verb is in the future tense or the first person since these are promises about the future and not verifiable. Journalists found, however, that these may also be interesting for their long-term, issue-centric work (Juneja and Mitra, 2022). Thus, StatCheck allows them to turn the future and first-person filters on and off.

6.2 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. 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* (Arslan et al., 2020). We leveraged the ClaimBuster dataset (Arslan et al., 2020), containing check-worthy claims in English from the U.S. Presidential debates, to train a cross-lingual language model, XLM-R (Conneau et al., 2019), which can perform zero-shot classification on French sentences after training on English data.

The ClaimBuster dataset ClaimBuster is a crowd-sourced dataset where the sentences from the 15 U.S. presidential elections debates from 1960 to 2016 have been annotated. The labels are Non-Factual Sentences (NFS), Unimportant Factual Sentences (UFS) or Check-Worthy Factual Sentences (CFS). The dataset contains $23K$ sentences, and the authors produced a subset of higher quality of $11K$ sentences for training models on classification tasks. In this smaller dataset, the NFS and UFS labels are grouped as negative labels, and the CFS labels are considered positive. We chose this higher-quality dataset to fine-tune the XLM-R model.

Fine-tuning the XLM-R model The XLM-R model is a Transformer-based masked language model trained on one hundred languages with 2.5TB of Common Crawl data. It achieves state-of-the-art results on multilingual tasks such as the XNLI benchmark (Conneau et al., 2018), while remaining competitive on monolingual tasks. We used a pretrained model with a vocabulary size of $250K$, 12 hidden layers of size 768 and 12 attention heads. 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 $80\%/\%10\%/10\%$.

Dataset	P	R	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.

Models	P	R	F1 score
OriginalStatClaim	0.692	0.466	0.557
StatClaim	**0.833**	0.517	0.638
CheckWorthyClaim	0.701	**0.915**	**0.794**

Table 2: Model evaluation on verifiable numerical claims.

Models	P	R	F1 score
OriginalStatClaim	0.282	0.688	0.400
StatClaim	**0.333**	0.750	**0.462**
CheckWorthyClaim	0.195	**0.938**	0.323

Table 3: Model evaluation on INSEE statistical claims.

Evaluation To optimize the performance, we trained the model with different hyperparameters. The best results were obtained with a learning rate of $5 \cdot 10^{-5}$, a batch size of 64, and using the AdamW optimizer. To evaluate the performance of the different models on French data, we annotated 200 randomly sampled French tweets and labeled them as check-worthy or not following the definition in (Arslan et al., 2020). Two annotators labeled each tweet; in the golden standard, a tweet is deemed check-worthy if both annotators agree on it, and not check-worthy otherwise. The Cohen Kappa score for inter-annotator agreement is 0.6, which is considered moderate to substantial agreement. The results can be found in Table 1. The performance on this test set is encouraging, however lower than on the original English dataset. This is expected given the zero-shot setting, as the tweets' format and vocabulary might differ from the ones in the training dataset.

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

6.3 Integration and Evaluation of the Claim Detection Models

We evaluate the claim detection models, *(OriginalStatClaim* (Duc Cao et al., 2019), *StatClaim* and *CheckWorthyClaim)*, on a set of 1595 tweets. Each tweet was labeled with two classes: *"Verifiable numerical claim"* (True if the tweet contains at least one numerical and verifiable claim") and *"INSEE statistical claim"* (True if the tweet contains at least one numerical, statistical claim verifiable against the INSEE dataset"). We chose these two labels as the first one gives us an indication of the tweets that can be verified if we had unlimited access to resources, while the second class identifies the tweets verifiable in the setting in which we have access to only one resource. We gathered 1595 random tweets from our scraped dataset to construct our set. Then, we automatically detected if a tweet contained a numerical value, if not, the tweet was labeled as negative for both classes. After that first step, we manually labeled the remaining 101 tweets. Two annotators labeled each tweet, and the gold standard was chosen as True if both annotators agreed. For the class *"verifiable numerical claim"*, we obtained a Kappa inter-Annotator Agreement score of 0.917 (almost perfect agreement), and *59* tweets were labeled as positive. For the class *"INSEE statistical claim"* we obtained an inter-annotator Agreement score of 0.807 (substantial agreement) and *16* tweets were labeled as positive.

Evaluation procedure For StatClaim and OriginalStatClaim, a tweet is considered positive if models return at least one extracted statistical statement. Our StatClaim was used in its default configuration: extractions with numerical values and without verbs conjugated in the future or in the first person. For CheckWorthyClaim, a tweet is considered positive if the model returns a check-worthy score > 0.9. We report the results in Table 2 and Table 3. StatClaim performs better than the original at detecting INSEE verifiable claims, and CheckWorthyClaim vastly outperforms both models on the detection of numerical claims, as they are a subset of check-worthy sentences that the model was trained to detect.

Finally, we evaluate the performance of our model directly against the journalist authors' prior manual work. For example, during the 2022 French presidential debate, the journalist team highlighted 29 of the 1954 uttered sentences and fact-checked them. The XLM-R model, on the other hand, classifies 443 of these sentences as check-worthy, and 27 of the 29 sentences chosen by the journalists are correctly classified. In other words, our model reduces by 77% the number of sentences to consider while retaining 93% of the sentences the journalists actually want to fact-check, saving the journalists considerable time without them missing too many important claims.

Default claim detection strategy. By default, StatCheck uses StatClaim for statistical claim detection. However, given the good performance of CheckWorthyClaim on numerical claims, we allow users to switch to it, even if we might not be able to verify them against the reference datasets.

7 Conclusion and Perspectives

Fact-checking journalists need automated tools to help scale up their daily work. We developed the StatCheck tool, which allows the journalist authors to focus their attention directly on check-worthy statements falling into one of two overlapping classes: those that can be checked based on statistics from two major institutions; and those that human users find interesting, even if the

data to back up the checks is not present in the database. StatCheck is in daily use in the fact-checking team; Figure 3 illustrates its GUI.

Quantitative question answering based on open data is gaining interest (Ho et al., 2020, 2022). In our continuing collaboration, we will work to extend StatCheck with more multidimensional statistic datasets from national governments and international organizations.

Acknowledgements This work is partially funded by AI Chair SourcesSay project (ANR-20-CHIA-0015-01).

References

2022. Media audience survey. Online (anonymized for double-blind reviewing).

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

Rami Aly, Zhijiang Guo, Michael Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. Feverous: Fact extraction and verification over unstructured and structured information.

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

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

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

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

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

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

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

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

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

Eurostat. 2022. European geographic location dictionary. https://ec.europa.eu/eurostat/estat-navtree-portlet-prod/BulkDownloadListing?sort=1&file=dic%2Ffr%2Fgeo.dic.

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

Vinh Thinh Ho, Koninika Pal, Niko Kleer, Klaus Berberich, and Gerhard Weikum. 2020. Entities with quantities: Extraction, search, and ranking. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 833–836. ACM.

Vinh Thinh Ho, Daria Stepanova, Dragan Milchevski, Jannik Strötgen, and Gerhard Weikum. 2022. Enhancing knowledge bases with quantity facts. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 893–901. ACM.

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

Prerna Juneja and Tanushree Mitra. 2022. Human and technological infrastructures of fact-checking. In *Proceedings of the 2022 Conference On Computer Supported Cooperative Work And Social Computing (CSCW '22). Proceedings of the ACM on Human-Computer Interaction.*

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

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

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

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese bert-networks. In *EMNLP*.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

Mohammed Saeed and Paolo Papotti. 2021. Fact-checking statistical claims with tables. IEEE Data Engineering Bulletin.

Vikash Singh. 2017. Replace or retrieve keywords in documents at scale. *CoRR*, abs/1711.00046.

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

Andreas Vlachos and Sebastian Riedel. 2015. Identification and verification of simple claims about statistical properties. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2596–2601, Lisbon, Portugal. Association for Computational Linguistics.

W3C. 2013. SPARQL 1.1 query language. http://www.w3.org/TR/sparql11-query/.

www-rdf. 2022. Resource description format. https://www.w3.org/RDF/.

www-redis. 2022. Redis. https://redis.io/.

www-tdb2. 2022. Apache Jena TDB. https://jena.apache.org/documentation/tdb2/.

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

A Multidimensional Datasets Example

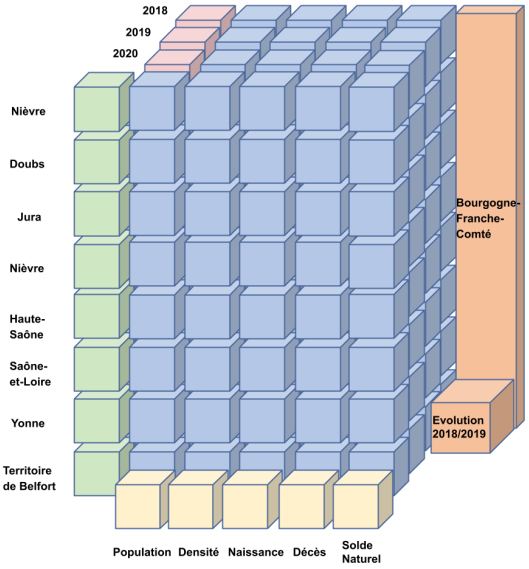
A.1 Insee

Figure 3 : Moins de naissances et plus de décès dans la moitié des départements
Bilan démographique en Bourgogne-Franche-Comté

	Estimation de population au 1er janvier 2020 (provisoire)	Densité de population au 1er janvier 2020 (habitant/km²)	Naissances domiciliées 2019	Naissances domiciliées Évolution 2018/2019 (%)	Décès domiciliés 2019	Décès domiciliés Évolution 2018/2019 (%)	Solde naturel 2019	Solde naturel 2018	Taux de natalité	Taux de mortalité brut	Taux d'accroissement naturel	Indicateur conjoncturel de fécondité	Espérance de vie des hommes à la naissance	Espérance de vie des femmes à la naissance
							En 2019 pour mille habitants						2019	
Côte-d'Or	532,500	61	5,060	- 2.5	4,920	- 1.1	140	220	9.5	9.2	+ 0.3	1.61	79.9	85.9
Doubs	543,000	104	5,760	- 3.9	4,720	- 2.2	1,040	1,170	10.6	8.7	+ 1.9	1.79	79.3	85.6
Jura	258,800	52	2,370	+ 3.1	2,790	- 0.6	-420	-510	9.2	10.8	- 1.6	1.87	80.0	85.1
Nièvre	201,800	30	1,520	- 8.1	2,990	- 2.0	-1,470	-1,400	7.5	14.8	- 7.3	1.81	77.1	83.9
Haute-Saône	234,600	44	2,040	- 3.2	2,480	+ 0.4	-440	-360	8.7	10.5	- 1.8	1.86	78.7	84.4
Saône-et-Loire	549,300	64	4,740	- 2.3	6,640	+ 2.0	-1,900	-1,600	8.7	12.1	- 3.4	1.85	79.1	85.0
Yonne	334,600	45	3,170	- 1.4	4,290	+ 1.8	-1,120	-990	9.5	12.8	- 3.3	1.98	77.5	83.9
Territoire de Belfort	139,900	230	1,450	- 4.6	1,360	+ 4.6	90	220	10.3	9.7	+ 0.6	1.82	78.6	84.0
Bourgogne-Franche-Comté	**2,794,500**	**58**	**26,110**	**- 2.7**	**30,190**	**+ 0.3**	**-4,080**	**-3,250**	**9.3**	**10.8**	**- 1.5**	**1.78**	**78.9**	**85.0**
France métropolitaine	**65,123,800**	**126**	**712,200**	**- 0.8**	**597,130**	**+ 0.5**	**115,070**	**123,500**	**11.0**	**9.2**	**+ 1.8**	**1.84**	**79.8**	**85.7**

Source : Insee, État civil, Recensements de la population, estimations de population

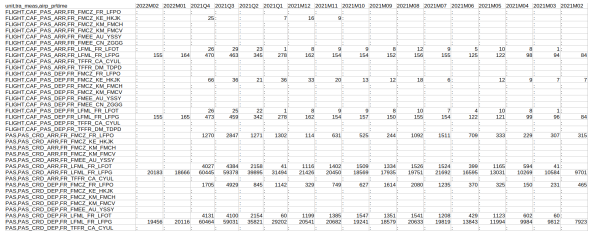
(a) An Insee table



(b) Insee multidimensional statistic dataset view

Figure 4: Example of Insee dataset

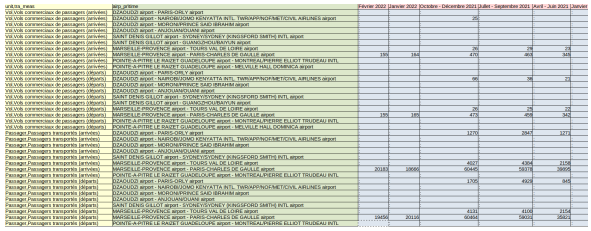
A.2 Eurostat



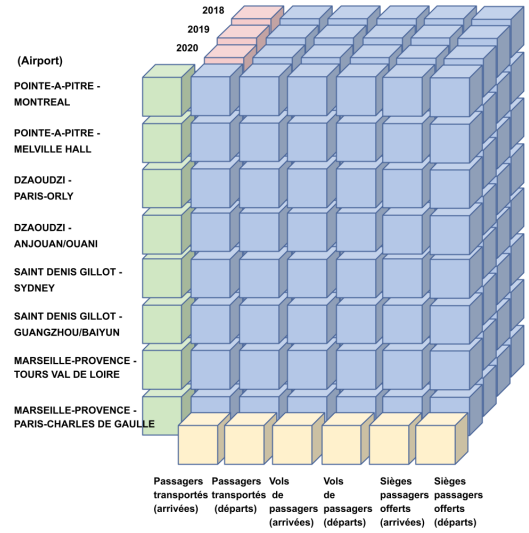
(a) An example of an encoded eurostat table

Statistical Code	Statistical Concept
FR_FMCZ_FR_FMEE	DZAOUDZI airport -SAINT DENIS GILLOT airport
FR_FMCZ_FR_LFML	DZAOUDZI airport -MARSEILLE-PROVENCE airport
FR_FMCZ_FR_LFPG	DZAOUDZI airport -PARIS-CHARLES DE GAULLE airport
FR_FMCZ_FR_LFPO	DZAOUDZI airport -PARIS-ORLY airport
FR_FMCZ_KE_HKJK	DZAOUDZI airport -NAIROBI/JOMO KENYATTA INTL. TWR/APP/NOF/MET/CIVIL AIRLINES airport
FR_FMCZ_KM_FMCH	DZAOUDZI airport -MORONI/PRINCE SAID IBRAHIM airport
FR_FMCZ_KM_FMCV	DZAOUDZI airport -ANJOUAN/OUANI airport
FR_FMCZ_MG_FMMI	DZAOUDZI airport -ANTANANARIVO/IVATO airport
FR_FMCZ_MG_FMNM	DZAOUDZI airport -MAHAJANGA/PH. TSIRANANA airport
FR_FMCZ_MG_FMNN	DZAOUDZI airport -NOSY-BE airport
FR_FMEE_AU_YSSY	SAINT DENIS GILLOT airport -SYDNEY/SYDNEY (KINGSFORD SMITH) INTL airport
FR_FMEE_CN_ZGGG	SAINT DENIS GILLOT airport -GUANGZHOU/BAIYUN airport
FR_FMEE_FR_FMCZ	SAINT DENIS GILLOT airport -DZAOUDZI airport
FR_FMEE_FR_FMEP	SAINT DENIS GILLOT airport -SAINT PIERRE PIERREFONDS airport
FR_FMEE_FR_LFBD	SAINT DENIS GILLOT airport -BORDEAUX-MERIGNAC airport
FR_FMEE_FR_LFBO	SAINT DENIS GILLOT airport -TOULOUSE/BLAGNAC airport

(b) An subsample of a dictionnary used to decode the encoded table (a)



(c) The decoded Eurostat table



(d) Eurostat multidimensional statistic dataset view

Figure 5: Example of Eurostat dataset