Reading and Reasoning over Chart Images for Evidence-based Automated Fact-Checking

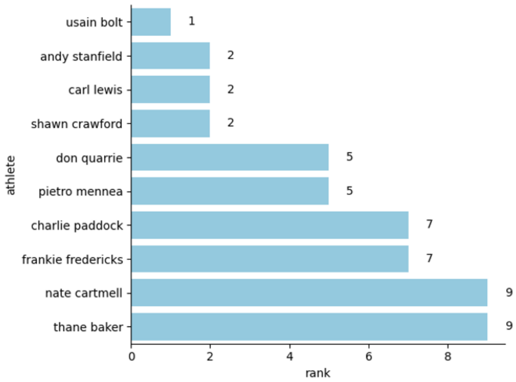
Mubashara Akhtar, Oana Cocarascu and **Elena Simperl**
Department of Informatics, King's College London
{mubashara.akhtar,oana.cocarascu,elena.simperl}@kcl.ac.uk

Abstract

Evidence data for automated fact-checking (AFC) can be in multiple modalities such as text, tables, images, audio, or video. While there is increasing interest in using images for AFC, previous works mostly focus on detecting manipulated or fake images. We propose a novel task, chart-based fact-checking, and introduce ChartBERT as the first model for AFC against chart evidence. ChartBERT leverages textual, structural and visual information of charts to determine the veracity of textual claims. For evaluation, we create ChartFC, a new dataset of $15,886$ charts. We systematically evaluate 75 different vision-language (VL) baselines and show that ChartBERT outperforms VL models, achieving 63.8% accuracy. Our results suggest that the task is complex yet feasible, with many challenges ahead.

Claim: Both Thane Baker and Nate Cartmell were ranked last.

Evidence:



Label: Supports

Figure 1: An example from the ChartFC dataset where the claim is supported by the evidence chart.

1 Introduction

Charts are often used to present data in news articles, reports, scientific publications, and across social media posts (Lo et al., 2022; Zhang et al., 2021). For example, in recent years, charts have been widely used to guide policymakers in deciding health policies and to communicate COVID information with the general public; a popular example is the coronavirus dashboard by Johns Hopkins University,[1] which was integrated in several websites (Perkel, 2020).

Misinformation can spread through charts in various ways. Previous works in data visualization have discussed how misleading chart design can cause misinformation (Lo et al., 2022). However, a more subtle form of misinformation occurs during chart interpretation (e.g. through invalid comparisons, framing correlation as causation, or spreading of misleading claims). To identify these misinformation types not only the stand-alone chart but the chart together with its message need to be considered jointly (Lo et al., 2022). In this work, we focus on verifying whether charts support or refute claims about them.

There has been substantial progress in automated fact-checking (AFC) in recent years, with a focus on verifying claims against text (Wang, 2017; Thorne et al., 2018; Schuster et al., 2021; Thorne et al., 2021; Diggelmann et al., 2020), table (Aly et al., 2021; Diggelmann et al., 2020; Chen et al., 2020a; Akhtar et al., 2022), and image (Yao et al., 2022; Zlatkova et al., 2019; Qu et al., 2022) evidence. Previous work has widely ignored claim verification against chart images. There are several challenges related to chart fact-checking as opposed to other evidence modalities: the structural information, text in charts, and location of text need to be considered jointly for chart understanding. Text plays a key role and is used, for example, as bar labels, chart titles, or in legends to explain the use of colors. Moreover, verifying claims against charts requires different reasoning

[1] https://coronavirus.jhu.edu/map.html

types, e.g. retrieving values, finding extremes, or calculating a sum.

To address these challenges, we propose the chart fact-checking task where, given a text claim and a chart, the goal is to classify if it *supports* or *refutes* the claim. We introduce ChartBERT as the first model for AFC against chart evidence comprising (i) an OCR-based reading component to extract text and structural information from chart images; (ii) a sequence generation component to process the extracted information; and (iii) an encoding component that extends the BERT architecture (Devlin et al., 2019) with three additional structural embeddings to jointly learn textual and structural representations of chart images.

Moreover, we release ChartFC as the first benchmark for chart-based AFC, created using TabFact (Chen et al., 2020a) as a seed dataset. Our dataset contains $15.9k$ human-written claims and bars of different colors, orientations, and backgrounds (see Figure 1 for an example). Our highest-performing ChartBERT model achieves 63.8% accuracy on ChartFC. We compare ChartBERT to 75 vision-language (VL) baselines, combining five vision encoders, three language encoders, and five fusion methods. The best-performing VL model is a transformer-based (Vaswani et al., 2017), dual encoder architecture that uses a simple, yet effective fusion block: concatenation and gated recurrent units (GRUs) (Bahdanau et al., 2015). Our results suggest that state-of-the-art VL approaches struggle with the proposed task, calling for more research.

Our **contributions** are as follows: 1) we propose the chart fact-checking task and build ChartBERT as the first chart fact-checking model; 2) we introduce ChartFC, the first dataset for AFC with chart evidence; 3) we systematically evaluate state-of-the-art language/vision encoders and fusion methods on the proposed task, highlighting challenges and providing an analysis of common reasoning types that contribute to failures.[2]

2 Related Work

2.1 Verifying Claims against Evidence

Evidence-based fact-checking aims to predict claims' veracity given evidence data. While many datasets focus on text (Thorne et al., 2018; Kotonya and Toni, 2020; Schuster et al., 2021; Wang, 2017) and table evidence (Chen et al., 2020a; Gupta et al., 2020; Aly et al., 2021; Wang et al., 2021a; Akhtar et al., 2022), human fact-checkers use a wider range of modalities for verification (Nakov et al., 2021b; Alam et al., 2021). They consult experts and extract information from databases, text, tables, graphics, and audio/video material from numerous sources.[3]

Charts influence how messages are perceived (Pandey et al., 2014). For example, Lee et al. (2021) use the term "counter-visualization" to describe data visualizations by the anti-vaccination communities in the US who created charts from publicly available data and interpreted them in a way that challenged the narrative of the pandemic, leading to disinformation.

2.2 Automated Fact-Checking with Images

Given that claims and evidence can be conveyed through different modalities, interest in AFC with images has increased recently (Nakov et al., 2021a; Cao et al., 2020; Alam et al., 2021; Yao et al., 2022; Sharma et al., 2022). Previous tasks focus mainly on detecting manipulated or fake images rather than on evidence-based claim verification (Blaier et al., 2021; Kiela et al., 2020; Alam et al., 2021; Sharma et al., 2022; Abdali, 2022). Whilst manipulated or fake images can be detected using the image only, claim verification requires understanding the claim and evidence jointly.

2.3 Chart Images in Other NLP Tasks

Two tasks related to chart fact-checking are chart question answering and chart summarization. Given a chart image, the summarization task requires to generate a summary of the chart in natural language text (Kantharaj et al., 2022; Tan et al., 2022). For question answering (chartQA) the answer to natural language questions is extracted from chart images. However, different to claim verification, questions typically provide strong indicators for the correct answers. Existing chartQA datasets are either small (Kim et al., 2020) or comprise automatically-generated, template-based questions (Chaudhry et al., 2020; Kahou et al., 2018; Kafle et al., 2018).

3 ChartBERT Model

We introduce ChartBERT, a first BERT-based chart fact-checking model. Our model consists of (i) a

[2] The ChartFC dataset, trained models, and our code are available at `github/link/to/chartfc.com`.

[3] `https://ballotpedia.org/The_methodologies_of_fact-checking`

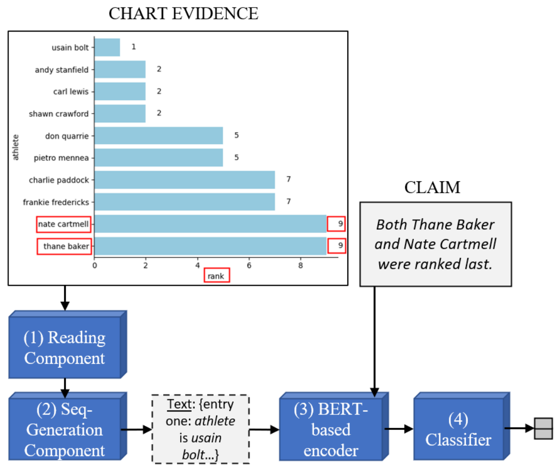


Figure 2: The ChartBERT architecture.

reading component which extracts text and structural information from charts (Section 3.2); (ii) a component for generating textual sequences from the information previously extracted (Section 3.3); and (iii) a BERT-based encoder with additional structural embeddings for the text extracted from charts (Section 3.4). The model architecture is shown in Figure 2.

3.1 Task Formulation

Following previous AFC work (Chen et al., 2020a; Aly et al., 2021; Thorne et al., 2018; Wang et al., 2021b), we view chart fact-checking as a classification task where, given a natural language claim and a piece of evidence (i.e. the chart image), the goal is to decide if the evidence *supports* or *refutes* the claim. We use support/refute as labels for claim classification instead of true/false as we only assess the claim veracity given the provided evidence rather than claiming universal statements.

Each ChartFC sample $i = (c_i, img_i, y_i)$ comprises a natural language claim c_i, a chart image img_i (see Figure 1 for an example), and a label $y_i \in \{supports, refutes\}$.

3.2 Reading Text from Chart Images

Given an image img_i, the reading component extracts text and structural information. First, we detect text regions in the chart using a Tesseract OCR model (Kay, 2007). Specifically, for each image, the model extracts n text regions $T_i = \{t_1, t_2, ..., t_n\}_{\text{j=1}}^{\text{n}}$, where each region t_j consists of $text_j$, a sequence of m tokens, and a rectangular bounding box b_j that surrounds the text region in the chart. The bounding box is a tuple $b_j = (x_j, y_j, w_j, h_j)$ where x_j and y_j are the pixel coordinates of the top left point of the box, and w_j and h_j represent the width and height of the box in pixels. Thus, for each image img_i we obtain the following output o_i:

$$o_i = f_R(img_i) = \{(text_j, x_j, y_j, w_j, h_j)\}_{\text{j=1}}^{\text{n}}$$

3.3 Text Sequence Generation

Next, we process the reading component's output into a text sequence s_i consisting of m tokens:

$$s_i = f_{SeqGen}(o_i) = [s_1, s_2, ...s_m]$$

We compare two approaches as follows.
Concatenation: The concatenation method processes the text regions (i.e. $t_j \in T_i$) based on their coordinates x_j and y_j so that texts that are close in the chart are also close in the generated sequence. The chart text is concatenated into one sequence and tokens that belong to different text regions are separated using a $[;]$ token. Thus, for the chart Figure 1 we obtain a text sequence starting with "usain bolt ; 1 ; andy stanfield ; 2 ; [...]."
Template: We use the structural information (i.e. x, y, w_j, h_j) to fill templates and generate text sequences. We evaluate three templates (an example for each template, extracted from Figure 1, is provided in brackets):
tmp_1: entry $[num]$: $[l_x]$ is $[text_x]$; $[l_y]$ is $[text_y]$ (entry one: athlete is usain bolt ; rank is 1);
tmp_2: "row $[num]$: $[l_x]$ is $[text_x]$; $[l_y]$ is $[text_y]$" ("row 0: athlete is usain bolt ; rank is 1");
tmp_3: "$[l_x]$ is $[text_x]$ when $[l_y]$ is $[text_y]$" ("athlete is usain bolt when rank is 1").

The placeholder $[l_x]$ is replaced with the x-axis label from the chart (e.g. "rank" in Figure 1). Similarly, the y-axis label (e.g. "athlete") replaces $[l_y]$. Based on the coordinates, we classify a bounding boxes that contain axes labels (i.e. the boxes with the largest y coordinates).

A counter starting from *one* replaces $[num]$ and numbers the bars in the chart. We fill $[text_y]$ and and $[text_x]$ with text regions detected as bar labels and axis ticks given their positions.

3.4 Encoding and Classification

ChartBERT captures the structure of charts through three learned embeddings: the *x coordinate embedding* which captures the horizontal location of the text in the chart, the *y coordinate embedding* which captures the vertical location, and the *label embedding* which takes value 1 if the text region is part

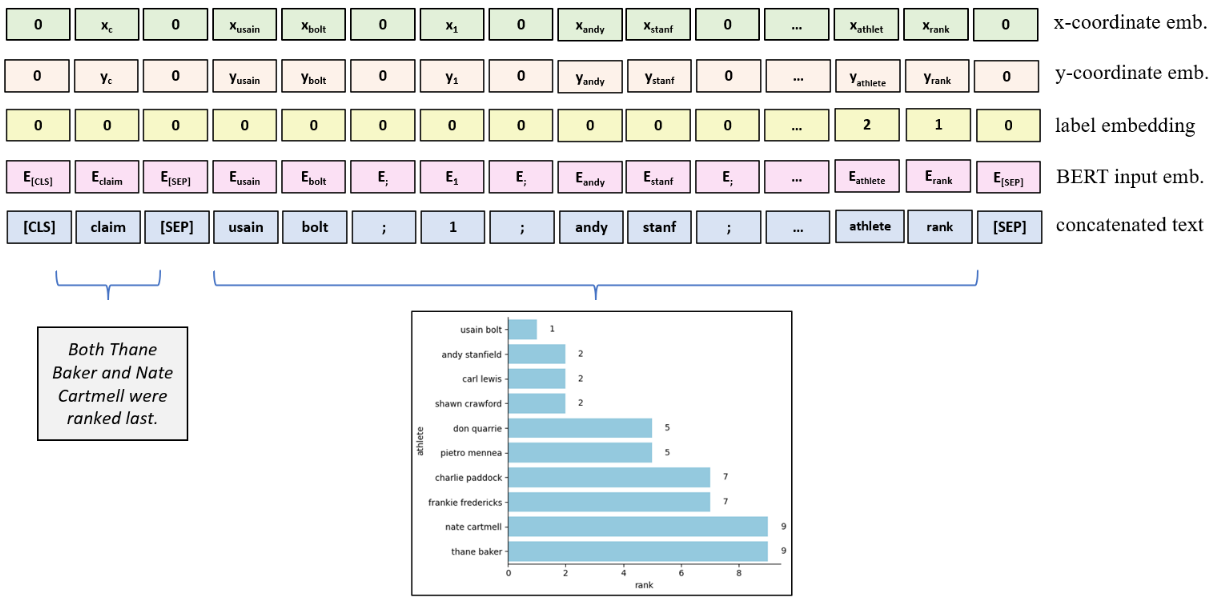

0 x_c 0 x_{usain} x_{bolt} 0 x_1 0 x_{andy} x_{stanf} 0 ... x_{athlet} x_{rank} 0
x-coordinate emb.
0 y_c 0 y_{usain} y_{bolt} 0 y_1 0 y_{andy} y_{stanf} 0 ... $y_{athlete}$ y_{rank} 0
y-coordinate emb.
0 0 0 0 0 0 0 0 0 0 0 ... 2 1 0
label embedding
$E_{[CLS]}$ E_{claim} $E_{[SEP]}$ E_{usain} E_{bolt} $E_;$ E_1 $E_;$ E_{andy} E_{stanf} $E_;$ ... $E_{athlete}$ E_{rank} $E_{[SEP]}$
BERT input emb.
[CLS] claim [SEP] usain bolt ; 1 ; andy stanf ; ... athlete rank [SEP]
concatenated text
Both Thane Baker and Nate Cartmell were ranked last.
usain bolt
andy stanfield
carl lewis
shawn crawford
don quarrie
pietro mennea
charlie paddock
frankie fredericks
nate cartmell
thane baker
1
2
2
2
5
5
7
7
9
9
athlete
0
2
4
6
8
rank

Figure 3: ChartBERT input representation with the text extracted from the chart and concatenated following the approach in Section 3.3. We include additional structural embeddings (i.e. x and y coordinates and label embeddings) to the BERT input embeddings (i.e. token, segment and position embeddings).

of the x-axis label (l_x), 2 if the text region is part of the for y-axis label (l_y) and 0 otherwise.

Figure 3 shows an example of the encoder with the structural embeddings. We concatenate claim c_i and sequence s_i, separate them with a $[SEP]$ token, add $[CLS]$ as the first input token, and feed the resulting vector as input to ChartBERT which generates 768-dimensional representations $h_i \in \mathbb{R}_{768}$. Finally, we pass h_i through a fully connected layer and determine the predicted label using sigmoid. ChartBERT uses binary cross entropy to minimize loss on the training set.

$$inp_i = (c_i, s_i, \{x_j, y_j, l^{\mathrm{x}}_{\mathrm{j}}, l^{\mathrm{y}}_{\mathrm{j}}\}^{\mathrm{n}}_{\mathrm{j=1}})$$

$$h_i = f_{Encoder}(inp_i)$$

$$p_i = \sigma(f_{FC}(h_i))$$

4 Evaluation

For evaluation, we first create a new dataset, ChartFC. We compare ChartBERT with several VL baselines, each comprising three components: a vision encoder, a language encoder, and a fusion block to obtain joint representations. We evaluate the dataset size and potential biases, discuss results obtained with ChartBERT and the baselines, and analyse reasoning types the models fail on.

4.1 ChartFC Dataset

This section provides an overview of the ChartFC dataset and its creation process. Each dataset entry comprises a natural language claim, a chart image, and a label $\in \{supports, refutes\}$.

4.1.1 The TabFact Dataset

We use TabFact (Chen et al., 2020a) as a seed dataset. TabFact is a table fact-checking dataset of natural language claims and tables extracted from Wikipedia as evidence, where the veracity of the claim is decided based on the accompanying table. Claims were written and evaluated by human crowdworkers with at least 95% approval rates for prior tasks and more than 500 accepted HITs on Amazon Mechanical Turk. The inter-annotator agreement for the claim verification task is *Fleiss* $\kappa = 0.75$.

4.1.2 Creation Pipeline

Figure 4 shows the dataset creation process.[4] Starting with $117,784$ claims and $16,000$ Wikipedia tables from TabFact, we first generate sub-tables. To link the claim text to table columns, we (i) lemmatize and tokenize the claim and the table content, (ii) link claim tokens to column headers and cells using string matching and heuristic rules, and (iii) decide if a claim token is linked to multiple columns using the minimum *Levenshtein distance*

[4] Figure 7 in the Appendix A illustrates the pipeline.

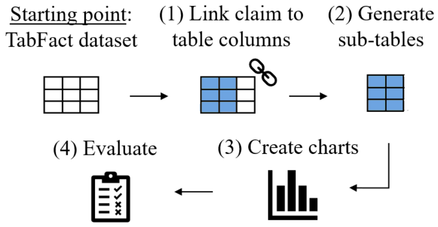


Figure 4: Dataset creation process.

	Train	Valid	Test	Sum
Support	7,048	896	885	8,829
Refute	5,654	697	706	7,057
Sum	12,702	1,593	1,591	15,886

Table 1: Class distribution across dataset split.

(Levenshtein, 1966), and finally, (iv) filter sub-tables with a maximum of twenty rows and two linked columns. This results in a total of $15,886$ pairs of claims and sub-tables.

Finally, we generate charts using the Python libraries *seaborn* and *matplotlib*. The charts vary across the dimensions (i) orientation (horizontal, vertical); (ii) bar colors (green, blue, pink); and (iii) background (no/white grid lines, white/gray background color). We show an example in Figure 1. We partition the dataset into training, validation, and test sets using 8:1:1 ratio and show statistics in Table 1.

4.1.3 Dataset Evaluation

To assess the data quality, we apply human and automated evaluation. We evaluate the sub-table generation step (step 2 in Figure 4) by checking the verifiability of claims against the extracted sub-tables with TableBERT (Chen et al., 2020a). We obtain 69.3% accuracy on our test set, comparable to 65.1% accuracy reported by Chen et al. (2020a) on their test set.

For human validation, we extract 100 random dataset entries and manually evaluate the claims against sub-tables and charts. Of the 100 claims, 92 were successfully verifiable against their sub-tables and chart images, six claims were not verifiable because a relevant column was missing in the sub-table, and two claims were already mislabelled in the TabFact dataset.

4.1.4 Chart Reasoning Types

We label 100 random test samples with *chart reasoning types*, using a taxonomy of common reasoning types humans apply while interacting with data visualisations (Amar et al., 2005). We find seven reasoning types present in our data: *retrieve value*, *filter*, *comparison*, *compute derived value*, *find extremum*, *determine range*, and *find anomalies*.[5] On average, we find 1.4 different types per claim with most claims including either one or two different reasoning types (see Figure 5). The reasoning type *retrieve value*, which requires extracting a value from the chart image given certain criteria, occurs most frequently (51%), followed by *find extremum*, i.e. highest or lowest values in the chart, and *filter*, which occur in approximately a quarter of all labelled claims. More complex types such as *compute derived value* or extracting all values in a given *range* are less frequent.

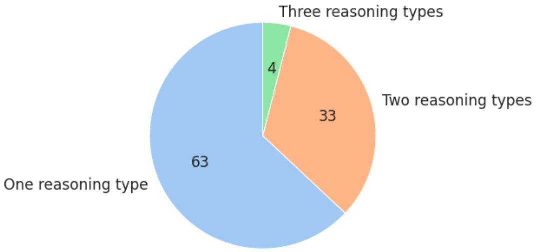


Figure 5: Number of chart reasoning types found in 100 dataset entries.

4.2 Vision-Language Baselines

We evaluate our task with several VL baselines, which jointly use claim text and visual information from images for claim verification. We also assess the top-3 VL baselines with OCR-extracted chart text as additional input. Each baseline consists of a language encoder, a vision encoder, and a fusion component to obtain joint representations. We systematically evaluate various state-of-the-art encoders and fusion techniques: we use shallow (e.g. BERT Embedder (Chen et al., 2020b)) and deep encoders (e.g. DenseNet (Huang et al., 2017)), as well as model-agnostic (e.g. concatenation) and model-based (e.g. transformer layers) fusion methods.

Language encoders: Given a claim c_i, we use a language encoder to obtain a feature vector:

$$h_i^{\text{text}} = f_{LangEncoder}(c_i)$$

We experiment with three language encoders:
BERT Embedder: Following Chen et al. (2020b), we tokenize the claim text into sub-words. For each token, we add the word and position embeddings to

[5] We describe the chart reasoning types in detail and give examples in Appendix B.

obtain the text representation which we then pass through a normalization (Ba et al., 2016) layer.

LSTM: We encode the text with 32-dimensional word embeddings and pass them through two LSTMs (Hochreiter and Schmidhuber, 1997) with 768-dimensional hidden states in each layer. We use the hidden states of the second layer as text representations.

BERT: We use a twelve-layer BERT encoder, initialized with weights from a pretrained BERT-base model.

Vision encoders: We use a vision encoder to extract representations for the chart images:

$$h_{\text{i}}^{\text{img}} = f_{VisEncoder}(img_i)$$

We evaluate five vision encoders:

Fully connected layer: We use a fully connected layer to extract 768-dimensional representations per image $h_{\text{i}}^{\text{img}} \in \mathbb{R}_{768}$.

AlexNet: Using AlexNet (Krizhevsky et al., 2012), for each image, we obtain a representation vector $h_{\text{i}}^{\text{img}} \in \mathbb{R}_{1024}$ by extracting the model output after the third max pooling layer.

ResNet: We use ResNet-152 (He et al., 2016) to obtain 2048-dimensional image representations by extracting the model output before the two final layers of ResNet-152, i.e. before the average pooling layer.

DenseNet: We use a DenseNet (DN) (Huang et al., 2017) comprising three dense blocks, with 6, 12, and 24 layers, respectively. We extract and concatenate the output of the first and third dense block as low- and high-level feature vectors: $h_{\text{i}}^{\text{img}} = f_{concat}(f_{DN[block1]}(img_i); f_{DN[block3]}(img_i))$.

Vision Transformer (ViT): We split images into sequences of n 16x16 patches before using them as input to a pretrained base-ViT model (Dosovitskiy et al., 2021).[6] We extract the hidden states from the model's final layer and use them as image representations, resulting in 768-dimensional vectors for each patch: $h_{\text{i}}^{\text{img}} = [h \in \mathbb{R}_{768}]_n$.

Fusion methods: We then fuse the text and image representations:

$$h_{\text{i}}^{\text{joint}} = f_{Fusion}(h_{\text{i}}^{\text{img}}; h_{\text{i}}^{\text{text}})$$

We experiment with five fusion methods:

Concatenation and multiplication: Concatenation and multiplication are common baseline approaches for multimodal fusion (Baltrušaitis et al., 2018). We reshape the text and image representations and either (i) concatenate both vectors, or (ii) perform element-wise multiplication.

Concatenation with GRUs: Inspired by Kafle et al. (2020), we concatenate the text and image representations and pass the resulting vector through m 1x1 convolutional layers and two GRUs. The first GRU takes the input in a forward direction, while the second GRU processes the input vector in a backwards direction to incorporate contextual information:

$$h_{\text{i}}^{\text{concat}} = f_{conv}(f_{concat}\{h_{\text{i}}^{\text{img}}; h_{\text{i}}^{\text{text}}\})$$

$$h_{\text{i}}^{\text{joint}} = f_{concat}\{f_{\overrightarrow{\text{GRU}}}(h_{\text{i}}^{\text{concat}}); f_{\overleftarrow{\text{GRU}}}(h_{\text{i}}^{\text{concat}})\}$$

Multimodal Compact Bilinear Pooling (MCB): MCB is an efficient and popular baseline for multimodal fusion (Fukui et al., 2016). The text and image representations are each projected to a higher dimensional space using the projection function Count Sketch (Charikar et al., 2004). The outer product of the projected vectors is then calculated in Fast Fourier Transform space to obtain a joint representation for both modalities and thus reduce the amount of learnable parameters during model training.

Transformer layers: Given the recent popularity of transformer layers used for joining text and visual representations (Tan and Bansal, 2019; Chen et al., 2020b; Yang et al., 2021), we use a three-layer transformer to get cross-modal embeddings.

The representation $h_{\text{i}}^{\text{joint}}$ is passed through two fully-connected layers and sigmoid to obtain the classification. We use binary cross entropy loss and stratified sampling in each training batch to minimize the loss on the training set.

4.3 Experimental Setup

We perform hyper-parameter search on the validation set and select the best-performing combination from the following values: $\{8, 16, 32\}$ for batch size, $\{1e^{-3}, 7e^{-4}, 5e^{-5}, 5e^{-6}, 5e^{-7}\}$ for learning rate, $\{1, ..., 50\}$ for training epochs with early stopping. We also experimented with different learning rates for the language and vision encoders. Ultimately, we used one learning rate for the entire VL model as the modality-specific learning rates did not provide any performance gains.[7]

[6] https://huggingface.co/google/vit-base-patch16-224

[7] The hyper-parameters for each VL baseline can be found in our GitHub repo.

SeqGen	Val Acc	Val $\mathbf{F_1}$	Test Acc	Test $\mathbf{F_1}$
concat.	59.2	55.1	60.6	57.0
temp. tmp_1	62.4	59.1	63.3	61.0
temp. tmp_2	62.0	59.4	61.9	58.7
temp. tmp_3	62.1	59.7	**63.8**	61.1

Table 2: Results for ChartBERT with different sequence generation (SeqGen) approaches: **concat**enation and **templ**ate.

V-Encoder	Fusion	no OCR	text concat
ViT	concat GRU	59.8	60.5
ResNet	mult	**60.1**	61.3
ResNet	concat	59.8	**62.7**

Table 3: Test accuracy of top-3 VL baselines: without (**no OCR**) chart text and chart **text concat**enated. All models use BERT as language encoder.

We run all experiments on a single NVIDIA Tesla V100 GPU with $32GB$ RAM. We measure model performance with prediction accuracy and (macro) F_1 on the test dataset.

4.4 Results & Discussion

How does ChartBERT perform on the task? How do different approaches for sequence generation influence model performance?

Table 2 gives an overview of the results obtained by ChartBERT. The best ChartBERT variant yields 63.8% test accuracy and processes chart text into text sequences using the template tmp_3. Compared to the concatenation approach, using tmp_3 increases the accuracy by $+3.2\%$.

Interestingly, the choice of template design impacts the model performance only slightly. While template tmp_3 might seem more "natural" to humans, it does not yield much higher performance compared to tmp_2.

How do VL baselines perform on ChartFC? How does the selection of encoder or fusion method impact model performance?

In contrast to many state-of-the-art VL approaches that use simple vision encoders and attention-based fusion (Chen et al., 2020b; Kim et al., 2021; Xia et al., 2021), the three best-performing VL models on ChartFC use BERT as language encoder, ViT or ResNet to obtain image representations, and either concatenation, multiplication, or concatenation with GRUs as a fusion method. Using only the claim and chart as input (i.e. without the OCR-extracted chart text), the highest test accuracy we obtain is 60.1% with the model consisting of BERT, ResNet, and multiplication fusion (see Table 3).

Regarding the language encoder,[8] models that use BERT perform best, irrespectively of the vision encoder and fusion method: the best LSTM-based model achieves 56.1% test accuracy and the best model with BERT embedder yields 56.5% accuracy, both lower than the best BERT-based VL model with 60.1% accuracy. In contrast, we obtain similar accuracy scores across different vision encoder: for example, replacing ResNet in Table 3 row two with a fully connected layer reduces the accuracy slightly by 0.6% to 59.7%. The choice of fusion method does not impact performance strongly: while using multiplication mostly outperforms other methods by a small margin, no fusion method stands out across all vision and language encoders. We also evaluate the chartQA model PReFIL (Kafle et al., 2020), which uses LSTM as language encoder, DenseNet for image representations, and concatenation with GRUs for fusion, and obtain on ChartFC a low test accuracy of 55.6%.

How does OCR-extracted chart text influence performance of VL models?

In addition to claim text and chart images used in VL baselines, we also include the text extracted from the charts through OCR as input (see Sections Sections 3.2 and 3.3 for details). Table 3 shows that using the concatenated chart text as input improves accuracy compared to the models that do no use the chart text (e.g. from 59.8% to 62.7%). The highest accuracy 62.7% is obtained with the BERT-ResNet-concatenation baseline.

Do models fail on particular chart reasoning types?

We evaluate the best VL baseline, consisting of BERT, ViT, and concatenation with GRUs, on the chart reasoning types present in ChartFC and described in Section 4.1.4. We find that the model performs best on the reasoning types *retrieve value*, *filter*, and *finding extremum*, while struggling particularly with *compute derived values*. Figure 6 shows that the model classifies correctly 65% (i.e. 33 out of 51) of claims that require *retrieval* and 61% of claims that require *filtering*. However, only 50% of *comparison* claims and 38% of claims required to *compute derived values* are correctly predicted. These results are in line with previous works that discuss limitations of state-of-the-art models in tasks requiring numerical reasoning capabilities (Thawani et al., 2021).

[8] The complete set of results obtained with different encoders and fusion methods can be found in Tables 5, 6, and 7 in the Appendix.

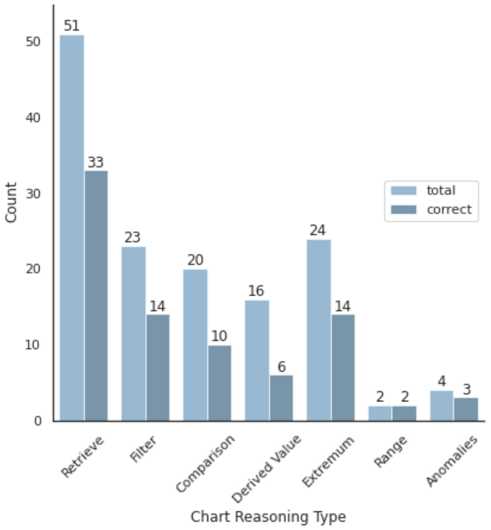


Figure 6: Chart reasoning types: total count and correct predictions of manually annotated test samples.

Training Samples	Test Accuracy
127 (1%)	51.6
3,175 (25%)	57.0
6,351 (50%)	57.1
9,526 (75%)	58.0
12,702 (100%)	59.8

Table 4: Performance of VL baseline (BERT, ViT, and concatenation with GRUs) with different training set sizes.

Is the dataset size sufficient for our proposed task? Do ChartFC claims contain biases?

We evaluate the size of the dataset by training our VL baseline (i.e. using BERT, ViT, and concatenation with GRUs) on various subsets of the training data as shown in Table 4 and report the accuracy on the test set. The performance on the test set improves as the number of training samples increases. While the performance gain is high when increasing the training set from 1% to 25% (51.6% accuracy compared to 57%), the difference in accuracy between the baseline trained on half of the training data and the entire training data is only 2.6%, indicating that our training set has a reasonable size.

We also train a claim-only BERT model to determine whether claims contain biases that allow the model to correctly predict the label while ignoring the evidence charts. Trained on the claim text only, the model achieves 52% accuracy on the test set, compared to ChartBERT's accuracy of (63.8%). We conclude that the claim text itself is not sufficient for correct classification.

What are the dis-/advantages of an automated dataset pipeline for chart fact-checking?

We automatically create ChartFC using a table fact-checking dataset as seed by identifying subtables relevant to the claims and then building the charts. ChartFC includes common stylistic variations: bars of different colors, horizontal/vertical orientations, different backgrounds (light/dark, grid lines/no grid lines). While natural charts come with large stylistic variation, using them results in reduced control over task complexity and dataset. In future work, we plan to explore two alternative dataset creation pipelines: first, automated pipelines for other charts types to extend the current dataset, and second, a pipeline with natural charts where we would create claims for charts.

Using natural charts would require a multi-step annotation process: selecting and separating charts from other images (Vougiouklis et al., 2020); writing claims which support/refute them; evaluating the claims to check for correctness, typos, etc. We would require annotators with proficiency in interpreting charts, and with basic mathematical and language skills to create claims with different reasoning types (see Figure 5).

5 Conclusion and Future work

We propose the chart fact-checking task and introduce ChartBERT, a novel model for fact-checking claims against chart images comprising three main components: a reading component, a sequence generation component, and an encoder that extends BERT's encoder with structural embeddings. We also introduce ChartFC as the first dataset for fact-checking against chart images, consisting of $15,886$ claims and chart images.

ChartBERT achieves 63.8% accuracy on ChartFC. We systematically evaluate 75 different VL baselines, using various language encoders, vision encoders, and fusion methods. The highest-performing VL baseline uses BERT as language encoder, ResNet to extract image representations, and concatenation to obtain joint representations for both modalities. The model achieves 62.7% test accuracy. Our results indicate that chart fact-checking, which requires extracting and reasoning over text and structural information from charts, is a challenging task for future research on AFC and VL methods.

Limitations

The TabFact dataset (Chen et al., 2020a) has been a valuable resource for creating ChartFC. However, using it as (the sole) seed dataset has limitations.

ChartFC consists of bar charts only; indeed, given the claims and tables found in TabFact, the bar chart was deemed the most appropriate chart type. Various types of charts exist (e.g. pie charts, line charts) and their effectiveness in different data contexts and tasks has been investigated in the literature. For example, Saket et al. (2019) evaluated the effectiveness of chart types using crowdsourcing experiments across the chart reasoning types we discussed in Section 4.1.4. In the context of small datasets, i.e. up to 34 rows and two columns which is similar to our setting, Saket et al. (2019) found bar charts to be the most accurate visualization type for the given chart reasoning types. In addition to bar charts, other types of charts used as evidence for fact-checking tasks ought to be investigated. Behrisch et al. (2018) studied visualization methods for different data types (i.e. multi- and high-dimensional data, relational data, geo-spatial data, sequential and temporal data, and text data). For example, they found that scatter plots were appropriate visualization types for queries regarding data distribution (e.g. correlations and clusters), while line charts were more appropriate for queries about temporal aspects of data. To extend ChartFC with other chart types, we require more diverse data types (e.g. sequential and temporal data) and appropriate claims.

Moreover, ChartFC claims are restricted to English, whereas misinformation is commonly spread in different languages. Future work is necessary to address the limited availability of non-English fact-checking datasets and to contribute to the efforts done in this space (Gupta and Srikumar, 2021).

References

Sara Abdali. 2022. Multi-modal misinformation detection: Approaches, challenges and opportunities. *CoRR*, abs/2203.13883.

Mubashara Akhtar, Oana Cocarascu, and Elena Simperl. 2022. PubHealthTab: A public health table-based dataset for evidence-based fact checking. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1–16, Seattle, United States. Association for Computational Linguistics.

Firoj Alam, Stefano Cresci, Tanmoy Chakraborty, Fabrizio Silvestri, Dimiter Dimitrov, Giovanni Da San Martino, Shaden Shaar, Hamed Firooz, and Preslav Nakov. 2021. A survey on multimodal disinformation detection. *CoRR*, abs/2103.12541.

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. FEVEROUS: fact extraction and verification over unstructured and structured information. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS*.

Robert A. Amar, James Eagan, and John T. Stasko. 2005. Low-level components of analytic activity in information visualization. In *IEEE Symposium on Information Visualization (InfoVis)*, pages 111–117. IEEE Computer Society.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR*.

Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2018. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):423–443.

Michael Behrisch, Michael Blumenschein, Nam Wook Kim, Lin Shao, Mennatallah El-Assady, Johannes Fuchs, Daniel Seebacher, Alexandra Diehl, Ulrik Brandes, Hanspeter Pfister, Tobias Schreck, Daniel Weiskopf, and Daniel A. Keim. 2018. Quality metrics for information visualization. *Comput. Graph. Forum*, 37(3):625–662.

Efrat Blaier, Itzik Malkiel, and Lior Wolf. 2021. Caption enriched samples for improving hateful memes detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9350–9358, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Juan Cao, Peng Qi, Qiang Sheng, Tianyun Yang, Junbo Guo, and Jintao Li. 2020. Exploring the role of visual content in fake news detection. *CoRR*, abs/2003.05096.

Moses Charikar, Kevin C. Chen, and Martin Farach-Colton. 2004. Finding frequent items in data streams. *Theor. Comput. Sci.*, 312(1):3–15.

Ritwick Chaudhry, Sumit Shekhar, Utkarsh Gupta, Pranav Maneriker, Prann Bansal, and Ajay Joshi. 2020. LEAF-QA: locate, encode & attend for figure question answering. In *IEEE Winter Conference on Applications of Computer Vision, WACV*, pages 3501–3510. IEEE.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020a. TabFact: A large-scale dataset for table-based fact verification. In *8th International Conference on Learning Representations, ICLR*. OpenReview.net.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020b. UNITER: universal image-text representation learning. In *Computer Vision - ECCV - 16th European Conference*, volume 12375 of *Lecture Notes in Computer Science*, pages 104–120. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Thomas Diggelmann, Jordan L. Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. 2020. CLIMATE-FEVER: A dataset for verification of real-world climate claims. *CoRR*, abs/2012.00614.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR*. OpenReview.net.

Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. 2016. Multimodal compact bilinear pooling for visual question answering and visual grounding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 457–468, Austin, Texas. Association for Computational Linguistics.

Ashim Gupta and Vivek Srikumar. 2021. X-fact: A new benchmark dataset for multilingual fact checking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 675–682, Online. Association for Computational Linguistics.

Vivek Gupta, Maitrey Mehta, Pegah Nokhiz, and Vivek Srikumar. 2020. INFOTABS: Inference on tables as semi-structured data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2309–2324, Online. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 770–778. IEEE Computer Society.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2261–2269. IEEE Computer Society.

Kushal Kafle, Brian L. Price, Scott Cohen, and Christopher Kanan. 2018. DVQA: understanding data visualizations via question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 5648–5656. Computer Vision Foundation / IEEE Computer Society.

Kushal Kafle, Robik Shrestha, Brian L. Price, Scott Cohen, and Christopher Kanan. 2020. Answering questions about data visualizations using efficient bimodal fusion. In *IEEE Winter Conference on Applications of Computer Vision, WACV*, pages 1487–1496. IEEE.

Samira Ebrahimi Kahou, Vincent Michalski, Adam Atkinson, Ákos Kádár, Adam Trischler, and Yoshua Bengio. 2018. Figureqa: An annotated figure dataset for visual reasoning. In *6th International Conference on Learning Representations, ICLR*. OpenReview.net.

Shankar Kantharaj, Rixie Tiffany Leong, Xiang Lin, Ahmed Masry, Megh Thakkar, Enamul Hoque, and Shafiq Joty. 2022. Chart-to-text: A large-scale benchmark for chart summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4005–4023, Dublin, Ireland. Association for Computational Linguistics.

Anthony Kay. 2007. Tesseract: an open-source optical character recognition engine. *Linux Journal*, 2007(159):2.

Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. The hateful memes challenge: Detecting hate speech in multimodal memes. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS*.

Dae Hyun Kim, Enamul Hoque, and Maneesh Agrawala. 2020. Answering questions about charts and generating visual explanations. In *CHI '20: CHI Conference on Human Factors in Computing Systems*, pages 1–13. ACM.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume

139 of *Proceedings of Machine Learning Research*, pages 5583–5594. PMLR.

Neema Kotonya and Francesca Toni. 2020. Explainable automated fact-checking for public health claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7740–7754, Online. Association for Computational Linguistics.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems*, pages 1106–1114.

Crystal Lee, Tanya Yang, Gabrielle D. Inchoco, Graham M. Jones, and Arvind Satyanarayan. 2021. Viral visualizations: How coronavirus skeptics use orthodox data practices to promote unorthodox science online. In *CHI '21: CHI Conference on Human Factors in Computing Systems, Virtual Event / Yokohama, Japan, May 8-13, 2021*, pages 607:1–607:18. ACM.

VI Levenshtein. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707.

Leo Yu-Ho Lo, Ayush Gupta, Kento Shigyo, Aoyu Wu, Enrico Bertini, and Huamin Qu. 2022. Misinformed by visualization: What do we learn from misinformative visualizations? *Comput. Graph. Forum*, 41(3):515–525.

Preslav Nakov, David P. A. Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barrón-Cedeño, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. 2021a. Automated fact-checking for assisting human fact-checkers. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI*, pages 4551–4558. ijcai.org.

Preslav Nakov, Giovanni Da San Martino, Tamer Elsayed, Alberto Barrón-Cedeño, Rubén Míguez, Shaden Shaar, Firoj Alam, Fatima Haouari, Maram Hasanain, Nikolay Babulkov, Alex Nikolov, Gautam Kishore Shahi, Julia Maria Struß, and Thomas Mandl. 2021b. The CLEF-2021 CheckThat! lab on detecting check-worthy claims, previously fact-checked claims, and fake news. In *Advances in Information Retrieval - 43rd European Conference on IR Research, ECIR*, volume 12657 of *Lecture Notes in Computer Science*, pages 639–649. Springer.

Anshul Vikram Pandey, Anjali Manivannan, Oded Nov, Margaret Satterthwaite, and Enrico Bertini. 2014. The persuasive power of data visualization. *IEEE Trans. Vis. Comput. Graph.*, 20(12):2211–2220.

J Perkel. 2020. Behind the Johns Hopkins University coronavirus dashboard. *Nature Index*, 7.

Jingnong Qu, Liunian Harold Li, Jieyu Zhao, Sunipa Dev, and Kai-Wei Chang. 2022. Disinfomeme: A multimodal dataset for detecting meme intentionally spreading out disinformation. *CoRR*, abs/2205.12617.

Bahador Saket, Alex Endert, and Çagatay Demiralp. 2019. Task-based effectiveness of basic visualizations. *IEEE Trans. Vis. Comput. Graph.*, 25(7):2505–2512.

Tal Schuster, Adam Fisch, and Regina Barzilay. 2021. Get your vitamin C! robust fact verification with contrastive evidence. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 624–643, Online. Association for Computational Linguistics.

Shivam Sharma, Firoj Alam, Md. Shad Akhtar, Dimitar Dimitrov, Giovanni Da San Martino, Hamed Firooz, Alon Y. Halevy, Fabrizio Silvestri, Preslav Nakov, and Tanmoy Chakraborty. 2022. Detecting and understanding harmful memes: A survey. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 5597–5606. ijcai.org.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Hao Tan, Chen-Tse Tsai, Yujie He, and Mohit Bansal. 2022. Scientific chart summarization: Datasets and improved text modeling.

Avijit Thawani, Jay Pujara, Filip Ilievski, and Pedro Szekely. 2021. Representing numbers in NLP: a survey and a vision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–656, Online. Association for Computational Linguistics.

James Thorne, Max Glockner, Gisela Vallejo, Andreas Vlachos, and Iryna Gurevych. 2021. Evidence-based verification for real world information needs. *CoRR*, abs/2104.00640.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pages 5998–6008.

Pavlos Vougiouklis, Leslie Carr, and Elena Simperl. 2020. Pie chart or pizza: Identifying chart types and their virality on twitter. In *Proceedings of the Fourteenth International AAAI Conference on Web and Social Media, ICWSM 2020, Held Virtually, Original Venue: Atlanta, Georgia, USA, June 8-11, 2020*, pages 694–704. AAAI Press.

Nancy X. R. Wang, Diwakar Mahajan, Marina Danilevsky, and Sara Rosenthal. 2021a. SemEval-2021 task 9: Fact verification and evidence finding for tabular data in scientific documents (SEM-TAB-FACTS). In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 317–326, Online. Association for Computational Linguistics.

Nancy Xin Ru Wang, Diwakar Mahajan, Marina Danilevsky, and Sara Rosenthal. 2021b. Semeval-2021 task 9: Fact verification and evidence finding for tabular data in scientific documents (SEM-TAB-FACTS). In *Proceedings of the 15th International Workshop on Semantic Evaluation, SemEval@ACL/IJCNLP*, pages 317–326. Association for Computational Linguistics.

William Yang Wang. 2017. "liar, liar pants on fire": A new benchmark dataset for fake news detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 422–426, Vancouver, Canada. Association for Computational Linguistics.

Qiaolin Xia, Haoyang Huang, Nan Duan, Dongdong Zhang, Lei Ji, Zhifang Sui, Edward Cui, Taroon Bharti, and Ming Zhou. 2021. XGPT: cross-modal generative pre-training for image captioning. In *Natural Language Processing and Chinese Computing - 10th CCF International Conference, NLPCC 2021, Qingdao, China, October 13-17, 2021, Proceedings, Part I*, volume 13028 of *Lecture Notes in Computer Science*, pages 786–797. Springer.

Zhengyuan Yang, Yijuan Lu, Jianfeng Wang, Xi Yin, Dinei Florêncio, Lijuan Wang, Cha Zhang, Lei Zhang, and Jiebo Luo. 2021. TAP: text-aware pre-training for text-vqa and text-caption. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 8751–8761. Computer Vision Foundation / IEEE.

Barry Menglong Yao, Aditya Shah, Lichao Sun, Jin-Hee Cho, and Lifu Huang. 2022. End-to-end multimodal fact-checking and explanation generation: A challenging dataset and models. *CoRR*, abs/2205.12487.

Yixuan Zhang, Yifan Sun, Lace M. K. Padilla, Sumit Barua, Enrico Bertini, and Andrea G. Parker. 2021. Mapping the landscape of COVID-19 crisis visualizations. In *CHI '21: CHI Conference on Human Factors in Computing Systems, Virtual Event / Yokohama, Japan, May 8-13, 2021*, pages 608:1–608:23. ACM.

Dimitrina Zlatkova, Preslav Nakov, and Ivan Koychev. 2019. Fact-checking meets fauxtography: Verifying claims about images. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2099–2108, Hong Kong, China. Association for Computational Linguistics.

A Dataset Pipeline

In Figure 7, we give an example of the dataset creation pipeline. Starting with the claim and initial TabFact table, we first filter columns required to decide the claims veracity label: "age at appointment" and "prior occupation". This sub-table is used to create the evidence chart (bottom right).

B Chart Reasoning Types

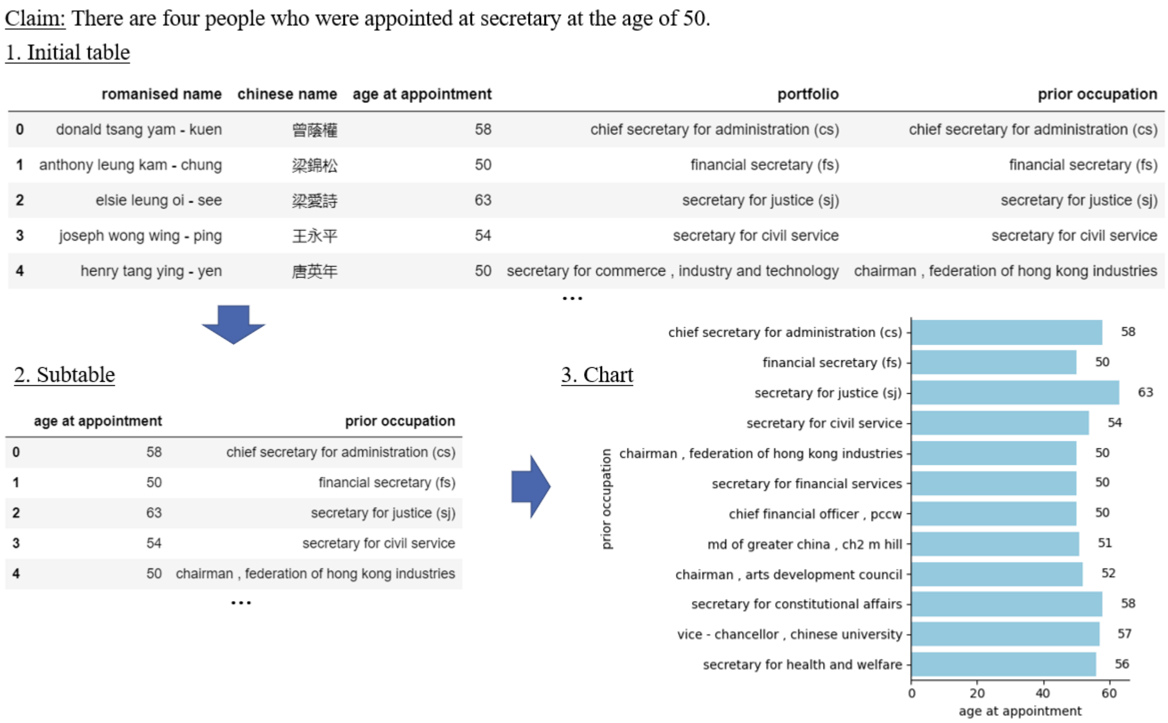
We label 100 random test set samples with chart reasoning types. Next, we briefly describe each type, for more details we refer to the taxonomy by Amar et al. (2005):

- Retrieve Value: Given some conditions, retrieve a single value from the chart image.
- Filter: Find all data points in the chart that fulfill some specified conditions.
- Compute Derived Value: Calculate an aggregated value (e.g. average or count) using data points extracted from the chart.
- Find Extremum: Extract the top-n data points given some conditions.
- Determine Range: Based on some conditions, find a span of values such that all extracted data points fulfil the conditions.
- Find Anomalies: Find any anomalies in a specified set of data points.
- Compare: Compare the values of different data points to each other.

C VL Baselines

Figure 8 provides an overview of all encoders and fusion methods we use in our evaluation.

Table 5, 6, and 7 provide an overview of all VL baselines we evaluated on ChartFC.



Claim: There are four people who were appointed at secretary at the age of 50.

1. Initial table

	romanised name	chinese name	age at appointment	portfolio	prior occupation
0	donald tsang yam - kuen	曾蔭權	58	chief secretary for administration (cs)	chief secretary for administration (cs)
1	anthony leung kam - chung	梁錦松	50	financial secretary (fs)	financial secretary (fs)
2	elsie leung oi - see	梁愛詩	63	secretary for justice (sj)	secretary for justice (sj)
3	joseph wong wing - ping	王永平	54	secretary for civil service	secretary for civil service
4	henry tang ying - yen	唐英年	50	secretary for commerce , industry and technology	chairman , federation of hong kong industries

...

2. Subtable

	age at appointment	prior occupation
0	58	chief secretary for administration (cs)
1	50	financial secretary (fs)
2	63	secretary for justice (sj)
3	54	secretary for civil service
4	50	chairman , federation of hong kong industries

...

3. Chart

chief secretary for administration (cs)
58
financial secretary (fs)
50
secretary for justice (sj)
63
secretary for civil service
54
chairman , federation of hong kong industries
50
secretary for financial services
50
chief financial officer , pccw
50
md of greater china , ch2 m hill
51
chairman , arts development council
52
secretary for constitutional affairs
58
vice - chancellor , chinese university
57
secretary for health and welfare
56
prior occupation
0
20
40
60
age at appointment

Figure 7: Example for dataset creation process.

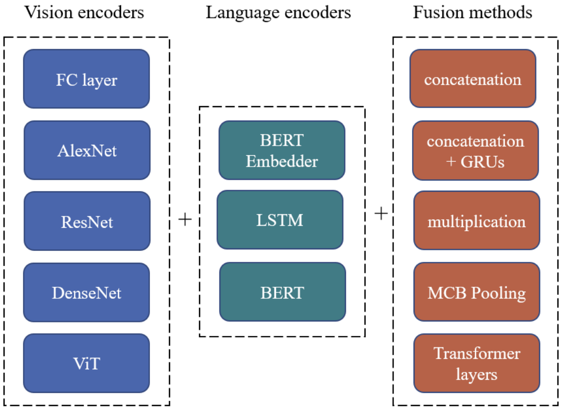


Figure 8: Encoders and fusion methods used in VL baselines.

Lang Encoder	**Vis Encoder**	**Fusion**	**Val Acc**	**Val F_1**	**Test Acc**	**Test F_1**
BERT Emb	FC	concatenation	56.7	37.8	55.6	36.6
BERT Emb	FC	concatenation, biGRU	56.2	36.0	55.6	35.7
BERT Emb	FC	multiplication	56.6	52.8	**56.5**	52.3
BERT Emb	FC	MCB	56.2	36.1	55.6	35.7
BERT Emb	FC	Transformer layers	56.2	36.0	55.6	35.7
BERT Emb	AlexNet	concatenation	56.5	40.2	55.1	38.1
BERT Emb	AlexNet	concatenation, biGRU	56.2	36.0	55.6	35.7
BERT Emb	AlexNet	multiplication	57.0	41.4	**55.9**	39.9
BERT Emb	AlexNet	MCB	56.2	36.0	55.6	35.7
BERT Emb	AlexNet	Transformer layers	56.2	36.0	55.6	35.7
BERT Emb	ResNet 152	concatenation	56.5	45.4	56.2	45.5
BERT Emb	ResNet 152	concatenation, biGRU	56.2	36.0	55.6	35.7
BERT Emb	ResNet 152	multiplication	56.6	38.3	**56.3**	38.8
BERT Emb	ResNet 152	MCB	56.2	36.0	55.6	35.7
BERT Emb	ResNet 152	Transformer layers	56.2	36.0	55.6	35.7
BERT Emb	DenseNet (6, 12, 24)	concatenation	56.5	43.7	54.0	40.7
BERT Emb	DenseNet (6, 12, 24)	concatenation, biGRU	56.6	45.3	54.1	42.2
BERT Emb	DenseNet (6, 12, 24)	multiplication	56.5	37.1	55.6	36.4
BERT Emb	DenseNet (6, 12, 24)	MCB	56.2	36.1	55.6	35.7
BERT Emb	DenseNet (6, 12, 24)	Transformer layers	56.2	36.0	55.6	35.7
BERT Emb	ViT	concatenation	56.2	36.0	55.6	35.7
BERT Emb	ViT	concatenation, biGRU	56.2	36.0	55.6	35.7
BERT Emb	ViT	multiplication	57.1	42.1	54.8	37.6
BERT Emb	ViT	MCB	56.2	36.0	55.6	35.7
BERT Emb	ViT	Transformer layers	56.2	36.0	55.6	35.7

Table 5: VL baselines using BERT embedder for text encoding, different vision encoders, and fusion methods

Lang Encoder	**Vis Encoder**	**Fusion**	**Val Acc**	**Val F_1**	**Test Acc**	**Test F_1**
LSTM	FC	concatenation	56.6	36.9	55.5	35.8
LSTM	FC	concatenation, biGRU	56.2	36.0	55.6	35.7
LSTM	FC	multiplication	56.2	36.0	55.6	35.7
LSTM	FC	MCB	56.2	36.0	55.6	35.7
LSTM	FC	Transformer layers	56.2	36.0	55.6	35.7
LSTM	AlexNet	concatenation	56.3	39.6	**56.1**	39.8
LSTM	AlexNet	concatenation, biGRU	56.2	36.0	55.6	35.7
LSTM	AlexNet	multiplication	56.2	36.0	55.6	35.7
LSTM	AlexNet	MCB	56.2	36.0	55.6	35.7
LSTM	AlexNet	Transformer layers	56.2	36.0	55.6	35.7
LSTM	ResNet 152	concatenation	56.2	36.0	55.6	35.7
LSTM	ResNet 152	concatenation, biGRU	56.2	36.0	55.6	35.7
LSTM	ResNet 152	multiplication	56.2	36.0	55.6	35.7
LSTM	ResNet 152	MCB	56.4	36.3	**56.0**	35.9
LSTM	ResNet 152	Transformer layers	56.2	36.0	55.6	35.7
LSTM	DenseNet (6, 12, 24)	concatenation	56.2	36.0	55.6	35.7
LSTM	DenseNet (6, 12, 24)	concatenation, biGRU	56.2	36.0	55.6	35.7
LSTM	DenseNet (6, 12, 24)	multiplication	56.2	36.0	55.6	35.7
LSTM	DenseNet (6, 12, 24)	MCB	56.2	36.0	55.6	35.7
LSTM	DenseNet (6, 12, 24)	Transformer layers	56.2	36.0	55.6	35.7
LSTM	ViT	concatenation	56.2	36.0	55.6	35.7
LSTM	ViT	concatenation, biGRU	56.2	36.0	55.6	35.7
LSTM	ViT	multiplication	56.2	36.0	55.6	35.7
LSTM	ViT	MCB	56.3	36.7	55.7	36.5
LSTM	ViT	Transformer layers	56.2	36.0	55.6	35.7

Table 6: VL baselines with LSTM as language encoder, different vision encoders, and fusion methods

Lang Encoder	**Vis Encoder**	**Fusion**	**Val Acc**	**Val F_1**	**Test Acc**	**Test F_1**
BERT	FC	concatenation	59.3	50.7	59.6	51.0
BERT	FC	concatenation, biGRU	58.8	51.1	58.5	50.2
BERT	FC	multiplication	59.4	54.5	**59.7**	54.9
BERT	FC	MCB	59.7	49.6	59.1	49.3
BERT	FC	Transformer layers	56.2	36.0	55.6	35.7
BERT	AlexNet	concatenation	59.5	47.9	59.1	47.6
BERT	AlexNet	concatenation, biGRU	59.2	48.2	58.0	47.0
BERT	AlexNet	multiplication	59.0	56.2	**59.6**	57.0
BERT	AlexNet	MCB	58.8	45.2	57.4	43.9
BERT	AlexNet	Transformer layers	57.6	50.8	59.5	52.6
BERT	ResNet 152	concatenation	59.8	50.9	59.8	50.8
BERT	ResNet 152	concatenation, biGRU	59.1	47.0	58.8	46.7
BERT	ResNet 152	multiplication	59.3	52.2	**60.1**	53.6
BERT	ResNet 152	MCB	58.2	47.0	58.7	48.9
BERT	ResNet 152	Transformer layers	56.2	36.0	55.6	35.7
BERT	DenseNet (6, 12, 24)	concatenation	59.1	51.4	**59.1**	52.4
BERT	DenseNet (6, 12, 24)	concatenation, biGRU	60.2	53.0	59.0	51.0
BERT	DenseNet (6, 12, 24)	multiplication	59.4	49.2	58.7	48.7
BERT	DenseNet (6, 12, 24)	MCB	59.9	49.6	58.8	48.6
BERT	DenseNet (6, 12, 24)	Transformer layers	58.7	48.0	58.1	46.8
BERT	ViT	concatenation	56.2	36.0	55.6	35.7
BERT	ViT	concatenation, biGRU	59.0	51.2	**59.8**	51.7
BERT	ViT	multiplication	58.0	42.7	56.6	41.1
BERT	ViT	MCB	59.2	49.5	59.2	49.6
BERT	ViT	Transformer layers	57.1	40.8	55.9	39.1

Table 7: VL baselines with BERT as language encoder, different vision encoders, and fusion methods