arXiv:2109.09824v5 [cs.CV] 15 Sep 2022

Well Googled is Half Done: Multimodal Forecasting of New Fashion Product Sales with Image-based Google Trends

Geri Skenderi[1], Christian Joppi[2], Matteo Denitto[2] and Marco Cristani[1,2]

[1]Department of Computer Science, University of Verona, Italy
{name.surname}@univr.it
[2]Humatics Srl, Verona, Italy
{name.surname}@sys-datgroup.com

September 16, 2022

Abstract

New fashion product sales forecasting is a challenging problem that involves many business dynamics and cannot be solved by classical forecasting approaches. In this paper, we investigate the effectiveness of systematically probing exogenous knowledge in the form of Google Trends time series and combining it with multi-modal information related to a brand-new fashion item, in order to effectively forecast its sales despite the lack of past data. In particular, we propose a neural network-based approach, where an encoder learns a representation of the exogenous time series, while the decoder forecasts the sales based on the Google Trends encoding and the available visual and metadata information. Our model works in a non-autoregressive manner, avoiding the compounding effect of large first-step errors. As a second contribution, we present VISUELLE, a publicly available dataset for the task of new fashion product sales forecasting, containing multi-modal information for 5577 real, new products sold between 2016-2019 from Nunalie, an Italian fast-fashion company. The dataset is equipped with images of products, metadata, related sales, and associated Google Trends. We use VISUELLE to compare our approach against state-of-the-art alternatives and several baselines, showing that our neural network-based approach is the most accurate in terms of both percentage and absolute error. It is worth noting that the addition of exogenous knowledge boosts the forecasting accuracy by 1.5% WAPE wise, revealing the importance of exploiting informative external information. The code and dataset are both available at `https://github.com/HumaticsLAB/GTM-Transformer`.

1 Introduction

Sales forecasting is one of the earliest and most required forecasting applications (Choi, Hui, & Yu, 2013; Beheshti-Kashi, Karimi, Thoben, Lütjenband, & Teucke, 2015): driven by economic and financial reasons, the ability to anticipate the needs and behavior of customers can make a big difference for commercial activity, especially when large volumes of goods need to be managed. While the forecasting of time series with a known historical past has been analysed extensively (R. J. Hyndman & Athanasopoulos, 2021; Lara-Benítez, Carranza-García, & Riquelme, 2021), very little attention has been paid to a much more practical and challenging scenario: the forecasting of new products, which the market hasn't seen before. In many cases, such forecasts are made in a judgmental manner (R. J. Hyndman & Athanasopoulos, 2021) by experts that essentially take into consideration the characteristics of the newly designed product along with information on what is trending right now in the market to make an educated guess.

In this paper, we propose a non-autoregressive Transformer (Vaswani et al., 2017) model dubbed GTM-Transformer, which tries to mimic this behavior by modeling the sales of new products based on information coming from several domains (modalities): the product image; textual descriptors of category, color and fabric; temporal information in the planned release date; and

exogenous information on the trending tendencies of the textual descriptors in the form of Google Trends. This last component is a crucial part of GTM-Transformer, since it introduces external information on item popularity into the reasoning. Intuitively, it models what people are interested in and proves important for forecasting performance.

While it has been already shown that Google Trends can be used to predict diverse types of time series (from real estate sales to inflation expectations) (L. Wu & Brynjolfsson, 2015; Bulut, 2018; Hand & Judge, 2012; Hamid & Heiden, 2015; Guzman, 2011; Bangwayo-Skeete & Skeete, 2015), their adoption to clothing sales forecasting has only been suggested in (Silva, Hassani, Madsen, & Gee, 2019) but never tried in practice, especially in a new product forecasting setting. Technically, we demonstrate that Google Trends are valuable when encoded appropriately. Thanks to the Cross-Attention weights of our model, we find that the most useful information is systematically located around the end of the previous year's same fashion season, i.e., seven to ten months before the product is planned for exposure.

As a second contribution, we present VISUELLE: the first public dataset for new fashion product sales forecasting. VISUELLE is a repository build upon the data of a real fast fashion company, Nunalie[1] and is composed of 5577 new products and about 45M sales related to fashion seasons from 2016-2019. Each product in VISUELLE is equipped with multimodal information: an image, textual metadata, sales after the first release date, and three related Google Trends describing *category*, *color* and *fabric* popularity. We use VISUELLE to compare GTM-Transformer with the few and recent state-of-the-art alternatives in the new product sales forecasting literature, obtaining the best performance on several forecasting metrics. We also show that the model can be enriched with attributes which are automatically inferred from the image, considering the widely-used Fashion IQ attributes (H. Wu et al., 2020), ameliorating the final performance.

The rest of the paper is organized as follows: the ensuing Sec. will provide a general overview of the literature around forecasting in fashion and new product sales forecasting. In Sec. 3, an overview of the VISUELLE dataset is given, showing the available information and how the dataset can be used for further research on this topic. Sec. 4 explains the methodological background and details behind GTM-Transformer. In Sec. 5, the experiments are thoroughly explained and finally, the conclusions are drawn in 6.

2 Related Work

2.1 New product sales forecasting

Tackling the new product sales forecasting with machine learning tools has very few precedent cases (Ekambaram et al., 2020; Singh, Gupta, Jha, & Rajan, 2019). The intuition followed in general is that new products will sell comparably to similar, older products; consequently, these models should be able to understand similarities among new and older products.

In (Singh et al., 2019), a variety of boosting algorithms (XGBoost, Random Forest) and Neural Networks (MLP, LSTM) are taken into account, fed with textual attributes related to category and colors, and merchandising factors such as discounts or promotions. Notably, they do not make use of image features or exogenous information. The most related work with ours is (Ekambaram et al., 2020), where the authors use an autoregressive RNN model that takes past sales, auxiliary signals like the release date and discounts, textual embeddings of product attributes, and the product image as input. The model uses soft-attention to understand which of the modalities is the most important to the sales. The model then embeds and combines all these attended features into a feature vector which is fed to a GRU (Cho et al., 2014) decoder and used to forecast the item sales. In contrast to our work, (Ekambaram et al., 2020) do not make use of a "true exogenous" signal such as the Google Trends, the model is based on internal information available in the data. Additionally, the autoregressive nature of RNNs creates prediction curves which have a very common shape across products. Unfortunately the dataset and the code is proprietary and was not released. Recent work has explored the use of exogenous signals in fashion forecasting, with (Joppi, Skenderi, & Cristani, 2022) showing that it is possible to build informative popularity signals that are helpful for forecasting. Our works adds to the literature by using available popularity signals from the web and a Transformer-based model to predict the sales in one shot, without autoregression.

[1] `http://www.nunalie.it.`

Figure 1: Examples of Images Per Category

2.2 Datasets for fashion forecasting

Publicly available datasets to forecast on fashion data take into account diverse applications, dissimilar from new product forecasting. The "Clothing, shoes and jewelry" dataset has been used in (Ni, Li, & McAuley, 2019; Al-Halah, Stiefelhagen, & Grauman, 2020) to forecast fashion styles, that is aggregates of products of multiple brands, in terms of popularity on Instagram. In our case the problem is different, since we are focusing on *single* products and not on groups of products, so we have definitely fewer data to reason on. In addition, we are *considering genuine sales data*, and not popularity trends. This makes our research more impactful on an industrial level. The Fashion Instagram Trends(Ma et al., 2020) adds geographical information to forecast trends in specific places.

3 The VISUELLE dataset

VISUELLE describes the sales between October 2016 and December 2019 of 5577 products in 100 shops of Nunalie, an Italian fast-fashion company funded in 2003. For each product, multimodal information is available, which will be detailed in the following subsections.

3.1 Image data

Each product is associated with an RGB image, of resolution which varies from 256 to 1193 (width) and from 256 to 1172 (height) with median values 575 (w) 722 (h) . Images have been captured in a controlled environment, in order to avoid color inaccuracies and potential biases in the predictions (Nitse, Parker, Krumwiede, & Ottaway, 2004). Each image portrays the clothing item on a white background, with no person wearing it. Additionally, a binary foreground mask is provided.

3.2 Text data

Each product has multiple associated tags, which have been extracted with diverse procedures detailed in the following, and carefully validated by the Nunalie team. The first tag is the *category*, taken from a vocabulary of 27 elements, visualized in Fig. 2a; the cardinality of the products shows large variability among categories, because naturally some of them (e.g. long sleeves) are popular year-round. This fact demonstrates on particular and challenging aspect of the dataset, which renders the learning of sales dynamics over different categories non-trivial. The *color* tag

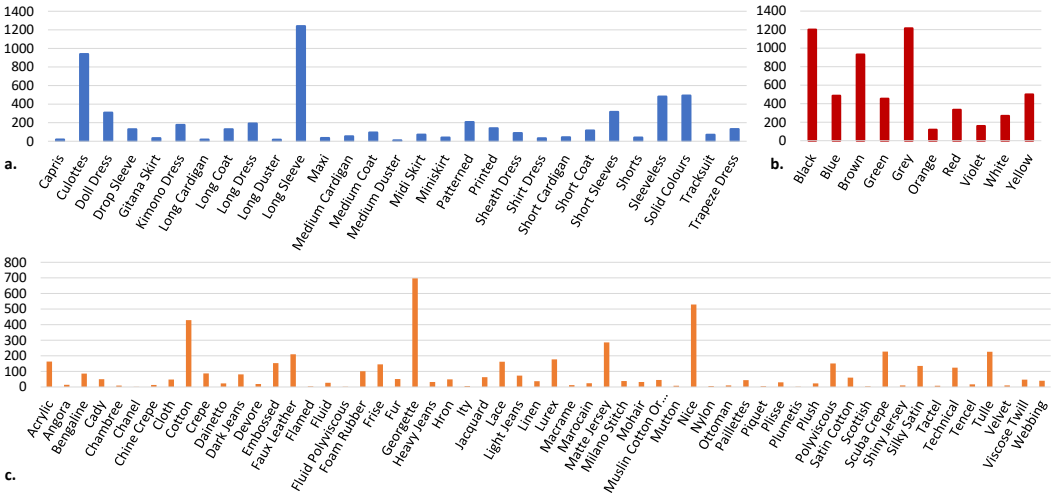

Figure 2: Cardinalities of the dataset per categories (a), color (b) and fabric (c)

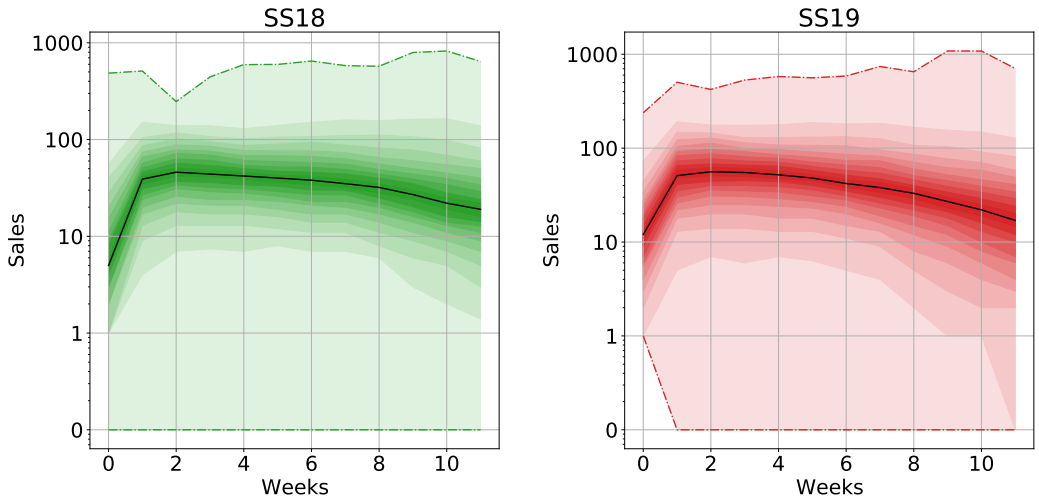

Figure 3: 25-percentile density plots of the SS18 and SS19 seasons.

represents the most dominant color, and is extracted from the images with a proprietary pixel clustering algorithm, keeping the color with the most belonging pixels, and validated for each product by two human operators that must agree on it. The final vocabulary is made of 10 elements. The cardinality per color is reported in Fig. 2b. The *fabric* tag describes the material from which clothes are made, and comes directly from the technical sheets of the fashion items. This tag comes from a vocabulary of 58 elements, visualized in Fig. 2c; A product is sold during a particular season, and within a season, released on the market at a precise day. This *temporal information* is recorded as a text string. Holidays and sales periods are supplementary information which we plan to deliver for a second version of the dataset.

3.3 Sales data

The sales time series have a weekly frequency and contain 12 observations each, which corresponds to the permanence of an item in the shops during a fashion season (Autumn-Winter, AW and Spring-Summer, SS). Fig. 3 contains a log-density plot of the sales of all the products, merging together different categories, across corresponding seasons (SS18 and SS19 were used for clarity). This is useful to show that there are general "mean curves" where the sales peak occurs after a week and that as the weeks go by, the sales are characterized by a higher variability. An increase

of the sales during the years is visible, showing that the company seems to perform well. Notably, from the release moment until 6 weeks, no external action is done by the company owners (discounts, pre/sales, additional supplying) and they had never sold out products, so we can state that the signal variability is given by the product attractiveness.

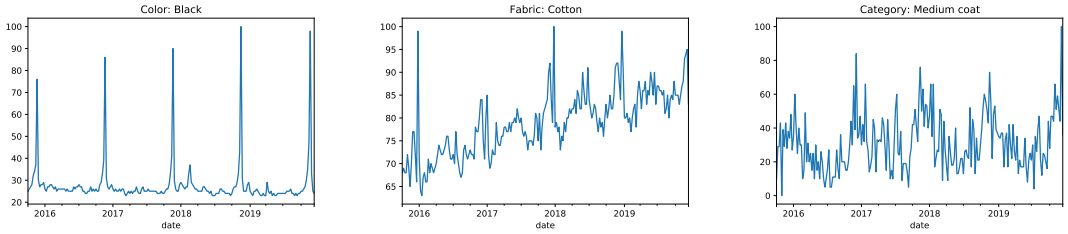

Figure 4: Examples of Google Trends time-series spanning multiple years.

3.4 Google Trends data

Extracting Google Trends to discover the popularity of textual term describing visual data poses a paradox: the more specific the text, the least informative the signal (due to sparsity), and vice-versa. In VISUELLE we collect, for each product, three Google trends time-series by querying the API using each of the product's three associated textual attributes: $color, category, fabric$. The trends are downloaded starting from the release date and going back 52 weeks, essentially anticipating the release of each single item by one year. Each signal represents a percentage, reaching 1 (100%) in the moment in time when that particular attribute had the maximum search volume on Google, for the duration of the search interval.

Fig.4 contains examples of Google Trends in the interval 2016-2019. As visible, the nature of these signals is highly variable, spanning from highly structured to more noisy. To make the Google trends signal more reliable, we follow the "multiple sampling" strategy discussed in (Medeiros & Pires, 2021). Google normalizes the search results of a query by the total searches of the location and time range chosen by the user. Then, the resulting numbers are scaled from 0 to 100, in order to represent the relative popularity. The problem is of course, because of the high amount of search queries that Google processes each day, the query results are always a sub-sample of the "true" ones and this sample may not always be the same. To avoid this occurrence, we download each Google Trend 10 times and use the mean to create a more representative signal.

4 Method

4.1 Background

Given a product x, we refer to its *product sales* time series as $S(x,t)$ where t refers to the t-th week of market delivery, with $x = 1, ..., N$ and $t = 1, ..., T$. In this work, we assume that each product x is associated with an image i, a set of textual tags T and a planned release date d, as in the VISUELLE dataset. Finally, exogenous series in the form of Google Trends are assumed to be available or collected as detailed in 3.4. The goal is to combine all of this information in an efficient and expressive way, such that we can forecast $S(x,t)$ as accurately as possible. We would like to stress again that in the new fashion product sales forecasting scenario, we cannot have direct access to $S(x,t)$, as the product x is new and therefore does not have past sales.

The structure of the proposed model is depicted in Fig. 5: GTM-Transformer is based on the Transformer model (Vaswani et al., 2017), but it deviates from the canonical form in two ways. Firstly, the decoder contains no self-attention block, since that the multimodal embedding that acts an input considering a non-autoregressive variant (Gu, Bradbury, Xiong, Li, & Socher, 2017), motivated by two reasons: i) to avoid the compounding of errors caused by wrong initial predictions; ii) to generate the forecasted time series in one go, without any recurrence mechanism, allowing for faster training and inference. In particular, GTM-Transformer learns different representations for each input type and then projects such representations in a shared latent space to non-autoregressively forecast the sales.

Before proceeding with the explanation of the model and its different components, we give a brief explanation of the attention mechanism, since it is the driving force behind the Transformer

model (Vaswani et al., 2017) and most state-of-the-art neural sequence processing tasks. An attention function can be described in simple words as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and final output are all vectors. The result is calculated as a weighted sum of the values, where the weight of each value is determined by the query's compatibility function with the corresponding key. These terms may sound confusing, but they are actually simple ideas coming from information retrieval systems. For instance, when we search for a query in a search engine, the engine will try to compare the *search query* to a list of *keys* (typically metadata) linked to potential matches in their database, and then return the *values* that give the best matches. To *learn* these query-key compatibilities, also known as attention weights, the Scaled Dot-Product Attention mechanism (Vaswani et al., 2017) projects the input in a common vector space and then relies on the dot-product to compute the similarity between vectors. The final output is actually a weighted average of the values according to the query-key compatibilities, because the learned attention weights are normalized to sum to one, as described by the following equations:

$$Q = W_q X + B_q, \tag{1}$$

$$K = W_k X + B_k, \tag{2}$$

$$V = W_v X + B_v, \tag{3}$$

$$\alpha = \frac{(QK^T)}{d_k}, \tag{4}$$

$$\hat{y} = softmax(\alpha)V, \tag{5}$$

where W and B are learnable weight and bias matrices for each transformation respectively, d_k is the dimensionality of the key(K) latent space, and the $softmax$ refers to the differentiable approximation of the $argmax$ function, which outputs a categorical probability distribution. By looking at equations 4 and 5, it becomes clear why the name of this method is "Scaled Dot-Product Attention". This powerful learning technique needs several training tricks to work well with sequences in practice, such as: Positional Encoding; Masking; Learning Rate warmp-up. For a detailed explanation of the Transformer model and its inner workings, we refer the reader to (Vaswani et al., 2017).

4.2 GTM-Transformer

Our architecture is an encoder-decoder sequence model (Sutskever, Vinyals, & Le, 2014) that is composed of several blocks, which interact with each other to create a model that is able to accurately forecast $S(x, t)$. To do this we rely mainly on dense vector representations of the different modalities and the Scaled Dot-Product Attention mechanism. The main idea behind the model is to use create a mixture between a retrieval system and a forecasting system, where the model first learns to generate an embedding for each product, and later performs a forecast. During the learning procedure, the model is able to learn and then reason on sales associations between similar and dissimilar products. In simpler words, we aim to build an architecture that is capable of learning (in a non-linear way) how to make judgemental forecasts (R. Hyndman & Athanasopoulos, 2018) of new fashion products. In order to do this, our model relies on its ability to retrieve similar items and understand their sales, as well as the popularity of the item tags that comes from the Google Trends. The building blocks of GTM-Transformer and their interactions are described below and depicted in Fig. 5.

4.2.1 Creating a multimodal product embedding

The first part of our architecture consists of a model that can effectively fuse the information coming from the different modalities. In order to this, we makes use of separate embedding modules for each modality, which create dense vector representations of the respective data in a shared latent space R^E. Afterwards, a Feature Fusion Network takes as input these vector representations and generates a unique, multimodal dense representation for the product.

The image embedding module relies on a ResNet-50 model (He, Zhang, Ren, & Sun, 2015) pre-trained on ImageNet (Deng et al., 2009) to extract 2D convolutional features $\phi_{resnet} \in R^{CxWxH}$, where $C = 2048$ is the number of final feature channels, W is the image width and H the image height. Adaptive average pooling (Liu, Qi, Qin, Shi, & Jia, 2018) with a square kernel of size 1 is applied, followed by a Fully Connected Layer, which generates the final image

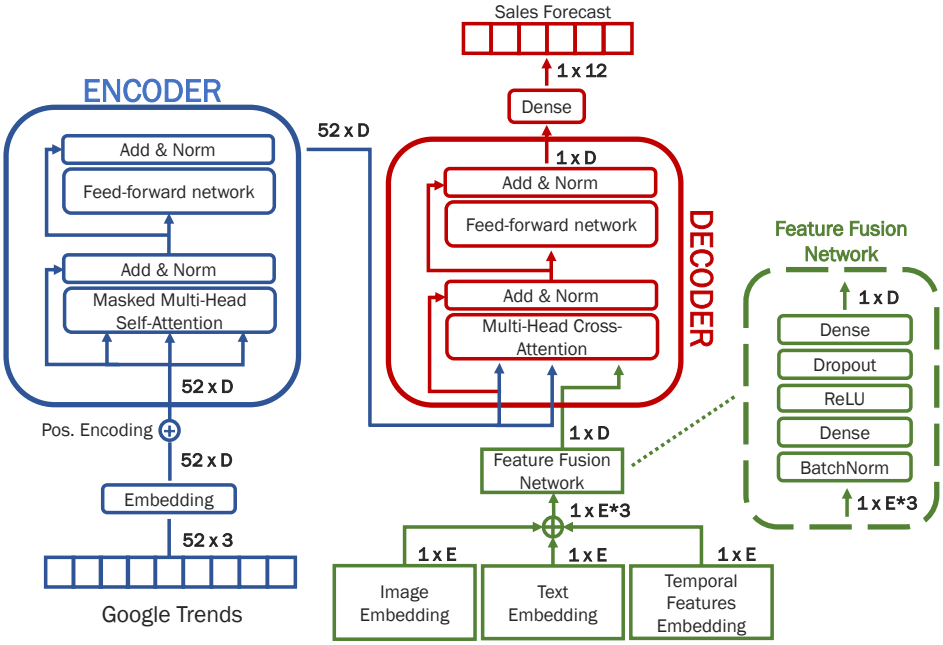

Figure 5: GTM-Transformer architecture. The encoder processes the exogenous Google Trends series and learns a representative embedding thanks to the self-attention mechanism. The decoder takes as input a multimodal embedding created from the Feature Fusion Network and then relies on a cross-attention mechanism to understand the implications of the Google Trend series on the multimodal embedding for the forecasting task. The output of the transformer model is then passed through a fully connected layer, to *generate* the sales forecast.

representation using the linear transformation:

$$\phi_i = W_i \phi_{resnet} + B_i,\ \phi_i \in R^E. \tag{6}$$

The text embedding module relies on the BERT model (Devlin, Chang, Lee, & Toutanova, 2019) pre-trained on a large corpus comprising the Toronto Book Corpus and Wikipedia. This module takes as input the same set of textual queries used to find the Google Trend (Sec. 3.4), i.e $color, category, fabric$ and produces an embedding $\phi_{bert} \in R^{768}$ of the words. Exploiting a pre-trained language model offers an additional advantage, which is the ability to generate a representation for any textual tag, even those that it might have never seen before. Because these language models are trained on extremely large corpora, they also add additional context that comes the typical uses of the textual tag in a sentence. The module averages the embeddings for each attribute and then uses a Fully Connected Layer to create the final textual representation:

$$\phi_t = W_t \phi_{bert} + B_t,\ \phi_t \in R^E. \tag{7}$$

The temporal features embedding module, is an MLP (Multi Layer Perceptron) that creates a set of embeddings $\{\phi_d, \phi_w, \phi_m, \phi_y\}$ that contains a projection of each temporal feature available for each product, extracted from the planned release date: the day of the week, the week of the year, the month and the year. Afterwards, these embeddings are concatenated and merged together through a dense layer, creating a final representation of all these temporal features:

$$\phi_r = W_r[\phi_d; \phi_w; \phi_m; \phi_y] + B_r,\ \phi_r \in R^E. \tag{8}$$

where $[;]$ is the concatenation operation.

The feature fusion network is another feed-forward network that merges the separate modalities together by using a cascade of fully-connected layers and a non-linear activation. This allows the model to generate an expressive and dense final product embedding in a higher dimensional space $R^D,\ D > E$, which also contains non-linear interactions between the different modalities:

$$\psi_f = f(\phi_i, \phi_t, \phi_r),\ \psi_f \in R^D \tag{9}$$

$$f(\phi_i, \phi_t, \phi_r) = W_{l2} ReLU(W_{l1}[\phi_i; \phi_t; \phi_r]) + B_{l2} \tag{10}$$

where $ReLU$ represents the Rectified Linear Unit activation function (Agarap, 2018).

4.2.2 Forecasting

GTM-Transformer first produces a self-attended representation of the Google Trend time series and then merges that information along with the previously explained multimodal embedding in order to forecast the future sales of a new product.

The transformer encoder takes as input a multivariate Google Trends time series (one for each textual attribute of the product). The series are projected in the same latent space R^D and then are enriched with a positional encoding. This signal is then processed by the standard encoder block of (Vaswani et al., 2017), by applying Scaled Dot-product Self-Attention as described in equations 1 - 5. We additionally employ *masking* with a block-diagonal matrix, which enforces locality (Rae & Razavi, 2020). The encoder outputs $\psi_g \in R^D$: a representation of the Google Trend time series enriched with information about which portions of itself are most important to the final exogenous representation for the forecasting task. This information is then fed to the decoder, acting as a prior of the popularity of the product.

The transformer decoder is the component which actually generates the forecasting. Unlike the vanilla decoder block of (Vaswani et al., 2017), we remove the Self-Attention segment, since the input coming from the Feature Fusion Network is a single representation and not a sequence. The input is fed to the Multi-Head Cross-Attention attention segment as the query, while ψ_g is projected as the key and query vector. Based on the explanation of the attention mechanism we provided in Sec. 4.1, this means that our model is trying to understand which part of the exogenous popularity series is most relevant to the multimodal embedding in order to generate a more accurate forecast. The decoder produces a final product embedding $\psi_f \in R^D$, which is a compact representation of four different modalities: $\{\psi_g, \phi_i, \phi_t, \phi_r\}$. Lastly, a Fully Connected layer projects ψ_p into $R^{horizon}$ in order to generate the forecasted time series $\{\hat{S}(x,1), ..., \hat{S}(x, horizon)\}$, based on the desired forecast horizon.

Training Our proposed architecture and all of its components are trained end-to-end using the Mean Squared Error (MSE) loss function, which is minimized using Mini-Batch Gradient Descent via Adafactor (Shazeer & Stern, 2018). The MSE for a batch of N items is defined as:

$$MSE(y, \hat{y}) = \frac{1}{N}\sum_{i=1}^{N}(y - \hat{y})^2 \tag{11}$$

Minimizing the square of the residuals is naturally a well-known and used method in regression problems, which also transfers to our work. By training our model to minimize this loss, we are trying to generate a point forecast that is as close as possible to the mean of the distribution of the sales process. It is widely known that one of the weak points of least-squares objectives is their sensitivity to outliers, which we try to overcome by performing pattern matching on the exogenous Google Trend signal. In this way, we are anticipating sale peaks by using prior online popularity, which in turn provides a data-driven approach that is both computationally cheap and does not rely on extremely complex modelling concepts.

5 Experiments

The experiments start in Sec. 5.1 with a preliminary study on how Google trends correlate with the sales. The experimental protocol and implementation details are explained in Sec. 5.2. In Sec. 5.3 we analyze the first results about how our approach performs against 9 comparative approaches covering the emerging literature of new product sales forecasting. Subsequently in Sec. 5.4, an ablation study investigates the role of the different modalities we take into account, namely textual data, image data and the Google trends (see Sec. 3). The analysis of the performance on the single categories is showed in Sec. 5.5, while the analysis on different time horizons completes the series of experiments in Sec. 5.6.

5.1 Correlation analysis with Google Trends

The goal is to check the strength and direction of monotonic association between the sales time series and the Google Trends, motivating their use in our framework. As a preprocessing step, we test the time series for stationarity using the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test (Kwiatkowski, Phillips, Schmidt, & Shin, 1992), to make sure that the potential correlations will

not be simply due to the temporal component (Aldrich, 1995). 34% of the sales time series are found to be non-stationary and are not considered for this initial analysis.

For each product, we utilize its associated 52-week Google Trends, based on the textual attributes. We calculate the Spearman correlation coefficient against the 12-week sales, using a sliding window protocol with window length $w = 12$ and stride of one step. Even though the small sample size does not encourage significance when performing correlation analysis (de Winter, Gosling, & Potter, 2016), we wish to investigate the distribution of significant correlations and in particular if they are located on specific periods of the trends. In other words, we are more interested in where the correlations are located across the trends, rather than their values.

The results give a statistically significant ρ coefficient in 86% of the total cases. On this selection, the strongest correlations were found to be positive, with 19% of all coefficients in the range [0.75,1]. The lags that contain the strongest correlations are contained mostly (54% of the cases) in the range [42,30], i.e., 42-30 weeks before the planned release date.

These findings are quite interesting, since they state that the period which is most correlated to the sales is seven to ten months before the product's release date, which corresponds loosely to the end of the same fashion season from the previous year. This preliminary analysis provides further motivation for the use of the Google Trends and is later confirmed by the cross-attention weights of GTM-Transformer in Sec. 5.7.

5.2 Experimental protocol

On VISUELLE we define an experimental protocol that simulates how a fast fashion company deals with new products, focusing on two particular moments: i) the *first order setup*, which is when the company orders the first stock of products to be distributed in the shops, usually two months before the starting season; ii) the *release setup*, which is right before the start of the season, and is useful to obtain the best forecast by using all of the exogenous information at hand, so to have a preliminary idea of when to do the stock replenishment. For these two moments we use 28 and 52 timesteps long Google trends, respectively.

As forecast horizon, we consider 6 weeks, as it is the period where no interventions are made by the company, such as reordering or retirements of products (if they perform very poorly). In any case, all models classifiers have been trained assuming a 12-week prediction, and shorter horizons have been taken into account for the evaluation. This procedure maximized the performances of all the approaches. Nonetheless results at different horizons will be shown here as for our approach. To perform the experiments, we divide the data into a training and testing partition, where the testing products are composed of the 497 most recent products. The rest of the dataset (5080 products) is used for training.

We utilize the *Weighted Absolute Percentage Error* (R. Hyndman, Koehler, Ord, & Snyder, 2008) as the primary error measure. It expresses the forecasting accuracy as a ratio:

$$\text{WAPE} = \frac{\sum_{t=1}^{T} |y_t - \hat{y}_t|}{\sum_{t=1}^{T} y_t} \tag{12}$$

where T is the forecasting horizon. WAPE is always nonnegative, and a lower value indicates a more accurate model. Even though it is a percentage-based metric, it is not bounded by 100.

For a more articulated understanding of our approach, we compute the *Mean Absolute Error* (MAE), also known as Mean Average Devation (MAD):

$$\text{MAE} = \frac{\sum_{t=1}^{T} |y_t - \hat{y}_t|}{T} \tag{13}$$

MAE describes the mean quantity by which the forecast misses the values on their respective scale.

Forecasting bias (Brown, 2004) is another aspect to take into account, measuring systematic over- or underestimation of the forecast w.r.t. the correct value. Even if a slight forecast bias might not have a notable effect on store replenishment, it can lead to over- or under-supply at the central warehouse. To measure the forecasting bias, we adopt the *tracking signal* (TS) measure (Brown, 2004; Nahmias & Cheng, 2009):

$$\text{TS} = \frac{\sum_{t=1}^{T} y_t - \hat{y}_t}{MAE} \tag{14}$$

which is basically the signed difference between actual and prediction value, divided by the MAE. The sign of the tracking signal communicates if we have an overestimation (if negative) or an underestimation (if positive). The closer to zero, the more unbiased the forecast. In the literature,

a forecasting approach is considered to be consistently biased if the tracking error is above 3.75 or below -3.75 (Brown, 2004; Nahmias & Cheng, 2009).

Finally, we focus on the capability in providing a forecasting curve which resembles the ground truth, as a way to highlight whether the model has properly captured the actual signal dynamics. To this end, we exploit the Edit distance with Real Penalty (ERP) (Chen & Ng, 2004) which borrows from the classical Edit Distance (ED). ED works on discrete sequences, counting the number of edit operations (insert, delete, replace) that are necessary to transform one series into the other. ERP uses the following algorithm: if the Euclidean distance between prediction $\hat{y}_t$ and y_t is smaller than a penalty ϵ, they are considered equal (d=0) and if not they are considered different (d=1). Summing over differences along the time axis gives the final distance. Since ERP is a dissimilarity, the closer it is to 0 the better.

5.3 Comparative results

Comparing GTM-Transformer with other approaches in the literature requires particular care, since we are the first to exploit Google Trends as exogenous variables to forecast sales for new products. For this reason, together with considering state-of-the-art alternatives in their original form, we adapt them by injecting Google Trends wherever this modification is natural, for example on models which already do process exogenous data. All the code, including the one for the competitors will be made publicly available, for the sake of fairness. To ease the reading, the name of the approaches will be followed by a square parenthesis indicating the type of information exploited within: T for textual data (category, color, fabric and release date), I for image data, G for google trends. Additionally, the name of the approaches which have been augmented with the Google Trends will be followed by a "+G". More in the detail, we consider:

kNN models. These non-parametric methods methods are proposed in (Ekambaram et al., 2020), and follow a common guideline for fast fashion companies: sales of new products will be similar to older, similar products they have already commercialized (Thomassey, 2014). The idea is to define a similarity metric between products and then forecast the sales of the new product by averaging the sales of the k most similar products that have sold before. Let P be set of all products and let $d(x_{p_i}, x_{p_j}), \forall x \in P$ be the distance between any two products. We can then obtain the set of k nearest neighbors to a product $K = \{x_1..x_k|P, d\}$. We can then estimate the sales of the a product x_p using a weighted average the sales of its neighbors $\sum_{k=1}^{K} \frac{d(x_p,x_k)}{\sum_{k=1}^{K} d(x_p,x_k)} y_k$, where y is the sales time series. The three KNN alternatives proposed in (Ekambaram et al., 2020) are all considered here, which depend on the data they consider to capture the similarity: i) between product attributes (color + category + fabric) *Attribute KNN*; ii) Between product images (*Image KNN*); iii) Between the product attributes *and* images *Attr. + Image KNN*. In our experiments, we use the cosine distance and set $k = 11$.

Gradient Boosting (GBoosting) (Friedman, 2001). This fundamental technique has been used in time series forecasting either as solitary models (Henzel & Sikora, 2020) and recently as components of more elaborate architectures (Ilic, Görgülü, Cevik, & Baydoğan, 2021). Gradient Boosting is an ensemble model which aggregate the results from multiple Decision Trees, where we assume Gradient Boosted Trees. Decision Trees are simple, tree-like diagrams for decision making. Gradient Boosted Trees build trees one after the other, such that each new tree helps correct the errors made by the previous one. This is done by fitting the trees on the negative of the gradient of a particular loss function (similarly to Backpropagation through SGD in Neural Networks). We use 500 trees and set least squares as the optimization problem. When using this model, the additional features, both exogenous and not, are concatenated together and fed to the model.

Multimodal Encoder-Decoder RNNs, proposed as most advanced techniques in (Ekambaram et al., 2020). The idea is to perform sequence learning in a two-step process, where an Encoder module takes the available information and produces a learned feature representation of the various modalities. This is then fed to an GRU(Cho et al., 2014) network that acts a Decoder, which autoregressively performs the forecasting. The authors augment their architecture with Bahdanau Attention(Bahdanau, Cho, & Bengio, 2016), using the last produced decoder hidden state to learn, at each prediction step, which one of the various modalities provides more important information to the forecast. In particular, we consider the two best performing techniques from the original paper, that is the *Concat Multimodal RNN* (Cat-MM-RNN), which which learns joint embeddings derived by concatenating embeddings of individual input modalities and the *Cross-Attention RNN* (X-Att-RNN), which learns multimodal attention weights and temporal attention weights to create an improved joint embedding. Both these architectures natively accomodate the use of Google Trends, so we feed the trends in the exogenous data module as depicted in (Ekambaram et al.,

Method	Input	6 Weeks, G.Trends: 52 weeks				6 Weeks, G.Trends: 28 weeks			
		WAPE	MAE	TS	ERP	WAPE	MAE	TS	ERP
Attribute KNN (Ekambaram et al., 2020)	[T]	59,8	32,7(18;39)	-0,88	0,40	59,8	32,7(18;39)	-0,88	0,40
ImageKNN (Ekambaram et al., 2020)	[I]	62,2	34,0(19;42)	-1,09	0,43	62,2	34,0(19;42)	-1,09	0,43
Attr+Image KNN (Ekambaram et al., 2020)	[T+I]	61,3	33,5(19;39)	-1,10	0,41	61,3	33,5(19;39)	-1,10	0,41
GBoosting (Friedman, 2001)	[T+I]	64,1	35,0(21;41)	-1,58	0,43	64,1	35,0(21;41)	-1,58	0,43
GBoosting+G (Friedman, 2001)	[T+I+G]	63,5	34,7(20;41)	-1,55	0,42	64,3	35,1(21;41)	-1,71	0,43
Cat-MM-RNN (Ekambaram et al., 2020)	[T+I]	63,3	34,4(18;44)	-0,67	0,42	63,3	34,4(18;44)	-0,67	0,42
Cat-MM-RNN+G (Ekambaram et al., 2020)	[T+I+G]	65,9	35,8(19;45)	-0,41	0,44	64,1	34,8(18;43)	-0,21	0,43
X-Att-RNN (Ekambaram et al., 2020)	[T+I]	59,5	32,3(16;39)	-0,32	0,38	59,5	32,3(16;39)	-0,32	0,38
X-Att-RNN+G (Ekambaram et al., 2020)	[T+I+G]	59,0	32,1(17;38)	-0,18	0,38	58,7	31,9(16;39)	-0,88	0,38
GTM-Transformer	[T+I+G]	55,2	30,2(15;36)	0,41	0,33	56,8	31,0(15;38)	0,90	0,35
GTM-Transformer **	[T+I+G]	54,2	29,6(14;35)	0,56	0,33	54,4	29,7(14;36)	0,44	0,31

Table 1: Results on VISUELLE. GTM-Transformer** is fed with the trends indexed by the tags automatically extracted from Fashion IQ. MAE is reported along with its values at the 25-th and 75-th percentiles in parenthesis.

2020). All neural network models are trained for 200 epochs with a batch size of 128 on an NVIDIA Titan RTX GPU.

Table 1 reports the results, where the following facts can be pointed out:

- The use of Google Trends boosts the performance of all the models, except Concat Multimodal RNN , where the Google Trends have been simply concatenated as static data.
- Our GTM-Transformer gives the best results in both setups (first order and release setup), with the best MAE and WAPE and the second best Tracking Signal, displaying a good balance between over and underestimation. We have the best ERP, which indicates that the shapes of our forecasting curves better resemble the actual sales (as also seen in Fig. 6).
- The tracking signal indicates persistent forecasting bias if its value is above (below) 3.75 (Brown, 2004; Nahmias & Cheng, 2009). Not one of the methods has this problem, including GTM-Transformer. This shows that even though the models have gotten much more complex, we are still able to maintain a strong balance between positive and negative errors. GTM-Transformer remains balanced even with 28-week Google Trends.
- Using shorter Google trends (28-week, Table 1 on the right) gives performances which in general are slightly worse. An explanation for this can be inferred when looking at the attention weights, which are explored in Sec. 5.7

To explore the generalization of the model to additional types of visual attributes, we consider the tags from Fashion IQs (H. Wu et al., 2020): they represent a widely-known approach to describe fashion items for automated retrieval purposes. We apply the attribute extraction code directly to our data, focusing on the "shape" attribute, which describes fine-grained aspects of the structure of the product (v-neck, hem, . . .). We discard the other types of attributes, since they consistently overlap with ours (such as the "fabric" attribute) or do not fit very well with VISUELLE, because in Fashion IQ clothes are worn by models. After the attribute extraction, we download the related Google Trends as described in Sec. 3. We dub this model in Table 1 as GTM-Transformer **. Interestingly, adding complementary information boosts further the model, promoting once again the use of the Google trends.

Additional insight can be inferred by some qualitative results, showing two 12-week predictions (Fig. 6): Attribute KNN gives reasonable estimates, trying to capture the scarce performance of the first 6 weeks portrayed in the second row. Gradient Boosting overestimates both the cases, offering a graphical demonstration of its high tracking signal TS=-1.58 (Table 1). The RNN-based approaches Concat Multimodal+G, Cross Attention RNN+G seems to have a very regular slope, irrespective of the real structure of the sale signal: this is likely due to the nature of the autoregressive approach, which has learned the general sale curve dynamics and struggles with trajectories which deviate from it. With the GTM-Transformer the role of the Google Trends appears to be clear, being capable of giving more structure to the final forecast (above), lowering down the forecasting thus predicting a scarce performance (below).

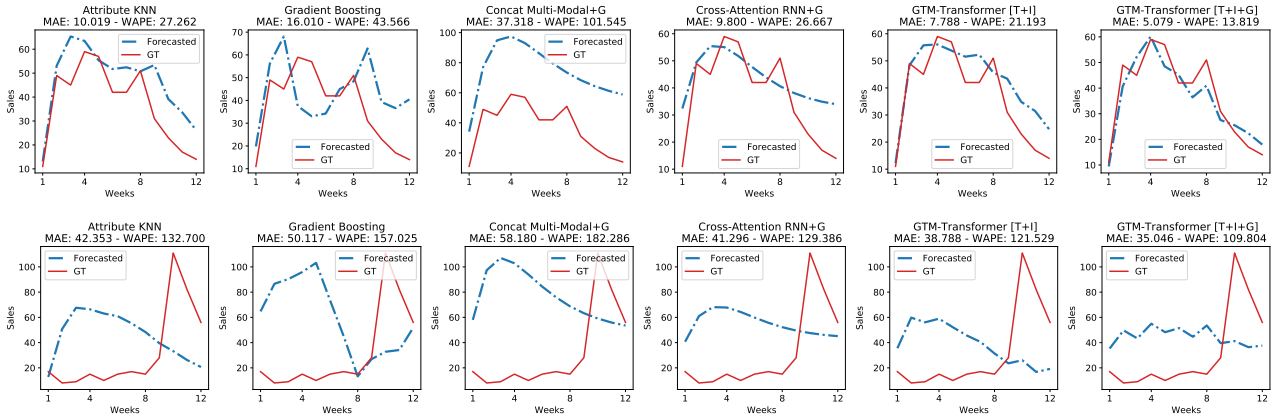

Figure 6: Qualitative Results

5.4 Ablation study

Ablative results refer to the 6-week forecasting horizon, using the full 52-week Google Trends, and are reported in Tab. 2.

GTM ablations	6 Weeks			
	WAPE	MAE (25%;75%)	TS	ERP
[I]	56,4	30,8(16;36)	-0,34	0,36
[T]	62,6	34,2(19;43)	-1,42	0,43
[G]	58,2	31,8(17;37)	-0,89	0,38
[I+T]	56,7	30,9(16;38)	-0,32	0,37
[T+G]	56,8	31,0(14;38)	1,63	0,33
[I+G]	55,7	30,4(13;32)	1,45	0,30
[T+I+G]	55,2	30,2(15;36)	0,41	0,33
[AR]	59,6	32,5(14;36)	1,18	0,32

Table 2: 6 weeks ablative results on VISUELLE. MAE is reported with also its values at 25-th and 75-th percentiles in parenthesis.

The first ablation is our model without the Google Trends, so removing the encoder module in Fig. 5 (row [I+T]). The much higher WAPE highlights the net role of the exogenous data, and is one of the main results of our study. It is worth noting that the performances are better than all of the approaches using the same kind of information (see Tab. 1), proving the good design of our architecture. The two-modality combos text + Google Trends ([T+G]) and image + Google Trends ([I+G]) give WAPE scores both around 57%, demonstrating that text and images carry complementary information which the complete GTM-Transformer is capable of combining and exploiting. Single modalities ablations instead demonstrate that the image alone [I] has the best performance, and this obviously states that it is the appearance of the product which allows for the most discrimination. Surprisingly, Google Trends [G] alone gives the second best results, while text attributes [T] alone gives the worst results, indicating once again the net value of this exogenous signal.

Finally, the *[AR]* row indicates the complete model, but in its autoregressive version: the performance is 4.4% worse than our GTM-Transformer, showing the benefit of the non-autoregressive design.

5.5 Single category analysis

Is interesting to check how GTM-Transformer performs on different categories. Figure 7 contains the separate WAPEs, where the marker size represents the cardinality of the category (Fig. 2a). The results confirm the fact that performances are more stable for categories with a large number of products such as "Long sleeve" or "Culottes", as the amount of data available for training over these products is larger.

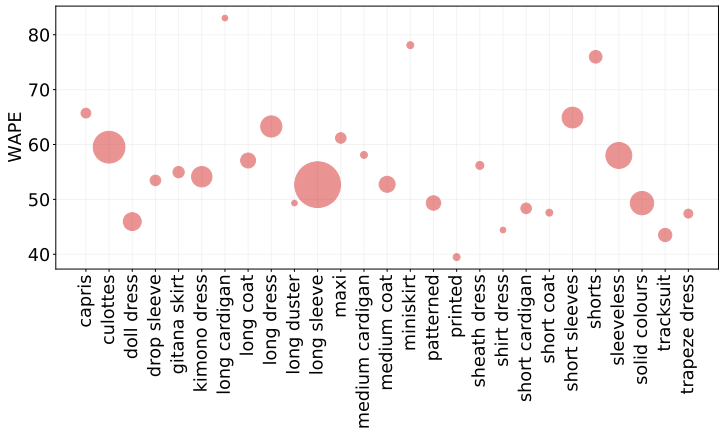

Figure 7: Category Results

5.6 Varying the forecasting horizon

In this section we demonstrate the effect of the forecasting horizon on the performance. Figure 8 contains the WAPE for 1, 2, 4, 6, 8 and 12 week forecasts. GTM-Transformer remains the best performing approach for all horizons, on pair at 2 weeks with Cross-Attention RNN+G. Most of the slopes show a minimum error at 6 weeks, except the Gradient Boosting which shows the second best performance at 1 week. The first 6 weeks performance varies greatly, with Attribute + Image KNN performing the worst. After 6 weeks, all the approaches have a decrease in the performance, which is natural, since the sale signal becomes more dependent on external choices (replenishments, discounts) we are not modeling here.

Figure 8: Different forecasting horizon results

5.7 Model interpretability: unveiling the Google Trends

To understand the role of Google Trends in GTM-Transformer we exploit the interpretability of the Attention mechanism. To this sake we calculate where in the Google Trend the decoder assigns the highest Cross-Attention weight for the testing set, to find if there are any systematical tendencies as to where the model looks at when making the prediction. A windown binning approach is used, similarly to the correlation analysis in Sec. 5.1, and we count the number of times that the lag with the highest weight is located in that particular window. Table 3 contains the results, where it can be seen that the initial period of the Google Trend seems to be the most crucial, as also hinted initially by our correlation analysis.

Lag	$52-40$	$42-30$	$32-20$	$22-10$	$12-0$
#Highest W	145	231	42	46	33

Table 3: Points of the Google Trends time series with the highest Cross-attention weights

5.8 A very practical use of our model: the *first-order* problem

Accurate new product forecasting is highly desirable for many reasons, as explained in the introduction: understand tendency in the sales, deciding when to replenish the warehouses, and how many products per reference to buy before the season starts. This is known as the *first-order* problem (Donohue, 2000), and it can be accurately simulated with the real data of VISUELLE. The goal is to order a number of products that matches the sum of future sales until the sixth week, without exceeding or underestimating. During the first six weeks then, sales will help with more

Method	6 Weeks	
	MAE	**US $ discr.** ↓
60% Policy	313,6	3.920.125 $
Attribute KNN (Ekambaram et al., 2020)	271,0	3.366.863 $
ImageKNN (Ekambaram et al., 2020)	279,7	3.475.242 $
Attribute + Image KNN (Ekambaram et al., 2020)	271,9	3.378.441 $
Gradient Boosting+G (Ilic et al., 2021)	297,2	3.692.453 $
Concat Multimodal KNN+G (Ekambaram et al., 2020)	359,7	4.495.977 $
Cross-Attention RNN+G (Ekambaram et al., 2020)	271,5	3.393.695 $
GTM-Transformer	**262,3**	**3.236.753** $

Table 4: First-order results on VISUELLE.

predictive power in suggesting how to behave with the remaining weeks, for example deciding whether to order again or not.

A general protocol to deal with the first order problem is to consider the sum of the sold products of the same period in the previous correspondent season, adding a percentage which mirrors the expected growth, and make the first order. In our case, the policy adopted by the company is to increase the orders for a product of a particular category, color and fabric by 60% of the previous average sum of sold products in the first six weeks for those attributes. We call this the 60% policy. For example, if we want to do the first order for SS19 season of a new white cotton cardigan, we take the average of sold white cotton cardigans of the previous SS18 and add the 60%.

To compute the first order error, we simply calculate the integral of the forecasting and ground truth curves for the first 6 weeks and compare them with each other, for each considered approach, including the 60% policy. To evaluate the performance, we compute the mean of all the absolute errors over all products. This tells us by how much, on average, the model is mistaken about the total sold amount and therefore the quantity of the first order. To show the real impact of such a problem, in Table 4 we report also the monetary discrepancy in US dollars, assuming that each reference has a cost of $25 (the average cost of a fast fashion product). In a market of around 12M dollars, the 60% policy is clearly ineffective, and all the forecasting approaches lower the discrepancy considerably, with GTM-Transformer lowering it the most.

6 Conclusion

In this paper we tackle the problem of new fashion product sales forecasting, which is a challenge that requires alternative solutions powered by machine learning and computer vision. In this scenario, we show that Google Trends are beneficial to forecasting and help augment the model's reasoning by adding a popularity prior for the items. All of this was possible thanks to a multimodal framework based on the Transformer, made non-autoregressive in order to deal with the complex dynamics which sales data exhibit, by effectively ingesting the Google Trends data. Additionally and thanks to the collaboration of Nunalie, a genuine dataset coming from the company's recent past sales has been proposed and made publicly available, equipped with ground truth sales signals and data from the image and text domain. Multiple directions can be considered for future work, starting from the use of a more "representative" popularity signal (data-centric aspect), to probabilistic modelling of the complex sales distributions of this practical and challenging problem.

Acknowledgements. This work has been partially supported by the project of the Italian Ministry of Education, Universities and Research (MIUR) "Dipartimenti di Eccellenza 2018-2022". We would like to thank Nunalie for their availability and for allowing us to publish VISUELLE.

References

Agarap, A. F. (2018). Deep learning using rectified linear units (relu). *CoRR*, *abs/1803.08375*. Retrieved from http://arxiv.org/abs/1803.08375

Aldrich, J. (1995). Correlations genuine and spurious in pearson and yule. *Statistical science*, 364–376.

Al-Halah, Z., Stiefelhagen, R., & Grauman, K. (2020, August). Fashion Forward: Forecasting Visual Style in Fashion. *arXiv:1705.06394 [cs]*. Retrieved 2020-10-25, from http://arxiv.org/abs/1705.06394

Bahdanau, D., Cho, K., & Bengio, Y. (2016). *Neural machine translation by jointly learning to align and translate.*

Bangwayo-Skeete, P. F., & Skeete, R. W. (2015). Can google data improve the forecasting performance of tourist arrivals? mixed-data sampling approach. *Tourism Management*, *46*, 454–464.

Beheshti-Kashi, S., Karimi, H. R., Thoben, K.-D., Lütjenband, M., & Teucke, M. (2015). A survey on retail sales forecasting and prediction infashion markets. *Systems Science and Control Engineering*, *3*, 154–161. doi: http://dx.doi.org/10.1080/21642583.2014.999389

Brown, R. G. (2004). *Smoothing, forecasting and prediction of discrete time series*. Courier Corporation.

Bulut, L. (2018). Google trends and the forecasting performance of exchange rate models. *Journal of Forecasting*, *37*(3), 303–315.

Chen, L., & Ng, R. (2004). On the marriage of lp-norms and edit distance. In *Proceedings of the thirtieth international conference on very large data bases-volume 30* (pp. 792–803).

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). *Learning phrase representations using rnn encoder-decoder for statistical machine translation.*

Choi, T.-M., Hui, C.-L., & Yu, Y. (2013). *Intelligent fashion forecasting systems: Models and applications*. Springer.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 ieee conference on computer vision and pattern recognition* (p. 248-255). doi: 10.1109/CVPR.2009.5206848

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). *Bert: Pre-training of deep bidirectional transformers for language understanding.*

de Winter, J. C., Gosling, S. D., & Potter, J. (2016). Comparing the pearson and spearman correlation coefficients across distributions and sample sizes: A tutorial using simulations and empirical data. *Psychological methods*, *21*(3), 273.

Donohue, K. L. (2000). Efficient supply contracts for fashion goods with forecast updating and two production modes. *Management science*, *46*(11), 1397–1411.

Ekambaram, V., Manglik, K., Mukherjee, S., Sajja, S. S. K., Dwivedi, S., & Raykar, V. (2020, August). Attention based Multi-Modal New Product Sales Time-series Forecasting. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 3110–3118). Virtual Event CA USA: ACM. Retrieved 2020-10-25, from https://dl.acm.org/doi/10.1145/3394486.3403362 doi: 10.1145/3394486.3403362

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, *29*(5), 1189 – 1232. Retrieved from https://doi.org/10.1214/aos/1013203451 doi: 10.1214/aos/1013203451

Gu, J., Bradbury, J., Xiong, C., Li, V. O., & Socher, R. (2017). Non-autoregressive neural machine translation. *arXiv preprint arXiv:1711.02281*.

Guzman, G. (2011). Internet search behavior as an economic forecasting tool: The case of inflation expectations. *Journal of economic and social measurement*, *36*(3), 119–167.

Hamid, A., & Heiden, M. (2015). Forecasting volatility with empirical similarity and google trends. *Journal of Economic Behavior & Organization*, *117*, 62–81.

Hand, C., & Judge, G. (2012). Searching for the picture: forecasting uk cinema admissions using google trends data. *Applied Economics Letters*, *19*(11), 1051–1055.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep residual learning for image recognition.*

Henzel, J., & Sikora, M. (2020). *Gradient boosting application in forecasting of performance indicators values for measuring the efficiency of promotions in fmcg retail.*

Hyndman, R., & Athanasopoulos, G. (2018). *Forecasting: Principles and practice* (2nd ed.). Australia: OTexts.

Hyndman, R., Koehler, A. B., Ord, J. K., & Snyder, R. D. (2008). *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media.

Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: principles and practice*. OTexts.

Ilic, I., Görgülü, B., Cevik, M., & Baydoğan, M. G. (2021). Explainable boosted linear regression for time series forecasting. *Pattern Recognition*, 108144.

Joppi, C., Skenderi, G., & Cristani, M. (2022). *Pop: Mining potential performance of new fashion products via webly cross-modal query expansion.* arXiv. Retrieved from `https://arxiv.org/abs/2207.11001` doi: 10.48550/ARXIV.2207.11001

Kwiatkowski, D., Phillips, P. C., Schmidt, P., & Shin, Y. (1992). Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of Econometrics*, *54*(1), 159-178. Retrieved from `https://www.sciencedirect.com/science/article/pii/030440769290104Y` doi: https://doi.org/10.1016/0304-4076(92)90104-Y

Lara-Benítez, P., Carranza-García, M., & Riquelme, J. C. (2021). An experimental review on deep learning architectures for time series forecasting. *CoRR*, *abs/2103.12057*. Retrieved from `https://arxiv.org/abs/2103.12057`

Liu, S., Qi, L., Qin, H., Shi, J., & Jia, J. (2018, June). Path aggregation network for instance segmentation. In *Proceedings of the ieee conference on computer vision and pattern recognition (cvpr).*

Ma, Y., Ding, Y., Yang, X., Liao, L., Wong, W. K., & Chua, T.-S. (2020, June). Knowledge Enhanced Neural Fashion Trend Forecasting. In *Proceedings of the 2020 International Conference on Multimedia Retrieval* (pp. 82–90). Dublin Ireland: ACM. Retrieved 2020-10-25, from `https://dl.acm.org/doi/10.1145/3372278.3390677` doi: 10.1145/3372278.3390677

Medeiros, M. C., & Pires, H. F. (2021). *The proper use of google trends in forecasting models.*

Nahmias, S., & Cheng, Y. (2009). *Production and operations analysis* (Vol. 6). McGraw-hill New York.

Ni, J., Li, J., & McAuley, J. (2019, November). Justifying recommendations using distantly-labeled reviews and fine-grained aspects. 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)* (pp. 188–197). Hong Kong, China: Association for Computational Linguistics. Retrieved from `https://www.aclweb.org/anthology/D19-1018` doi: 10.18653/v1/D19-1018

Nitse, P. S., Parker, K. R., Krumwiede, D., & Ottaway, T. (2004). The impact of color in the e-commerce marketing of fashions: an exploratory study. *European Journal of Marketing*.

Rae, J., & Razavi, A. (2020, July). Do transformers need deep long-range memory? In *Proceedings of the 58th annual meeting of the association for computational linguistics.* Online: Association for Computational Linguistics. Retrieved from

https://www.aclweb.org/anthology/2020.acl-main.672

Shazeer, N., & Stern, M. (2018). *Adafactor: Adaptive learning rates with sublinear memory cost.*

Silva, E. S., Hassani, H., Madsen, D. Ø., & Gee, L. (2019). Googling fashion: forecasting fashion consumer behaviour using google trends. *Social Sciences*, *8*(4), 111.

Singh, P. K., Gupta, Y., Jha, N., & Rajan, A. (2019, June). Fashion Retail: Forecasting Demand for New Items. *arXiv:1907.01960 [cs]*. Retrieved 2021-02-09, from http://arxiv.org/abs/1907.01960

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 27). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf

Thomassey, S. (2014). Sales forecasting in apparel and fashion industry: A review. *Intelligent fashion forecasting systems: Models and applications*, 9–27.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., … Polosukhin, I. (2017). *Attention is all you need.*

Wu, H., Gao, Y., Guo, X., Al-Halah, Z., Rennie, S., Grauman, K., & Feris, R. (2020). *Fashion iq: A new dataset towards retrieving images by natural language feedback.*

Wu, L., & Brynjolfsson, E. (2015). 3. the future of prediction: How google searches foreshadow housing prices and sales. In *Economic analysis of the digital economy* (pp. 89–118). University of Chicago Press.