Distributional Semantics of Line Charts for Trend Classification

Connor Onweller[1], Andrew O'Brien[2], Edward Kim[2], and Kathleen F. McCoy[1]

[1] University of Delaware, Newark, DE 19716, USA
{onweller,mccoy}@udel.edu
[2] Drexel University, Philadelphia, PA 19104, USA
{ao543,ek826}@drexel.edu

Abstract. Line charts are often used to convey high level information about time series data. Unfortunately, these charts are not always described in text, and as a result are often inaccessible to users with visual impairments who rely on screen readers. In these situations, an automated system that can describe the overall trend in a chart would be desirable. This paper presents a novel approach to classifying trends in line chart images, for use in existing chart summarization tools. Previous projects have introduced approaches to automatically summarize line charts, but have thus far been unable to describe chart trends with sufficient accuracy for real-world applications. Instead of classifying an image's trend via a convolutional neural network (CNN) system, as has been done previously, we present an architecture similar to bag-of-words (BoW) techniques for computer vision, mapping the image classification problem to an analogous natural language problem. We divided images into matrices of image patches which we then each treated as a series of "visual words" which were used to classify each image. We utilized natural language processing (NLP) word embeddings techniques to to create embeddings of visual words that allowed us to model contextual similarity between patches. We trained a linear support vector machine (SVM) model using these patch embeddings as inputs to classify the chart trend. We compared this method against a ResNet classifier pretrained on ImageNet. Our experimental results showed that the novel approach presented in this paper outperforms existing approaches.

Keywords: Assistive technology · Information graphic · Classification · Bag-of-visual-words · Distributional semantics.

1 Introduction

People with visual impairments often rely on screen-readers to perceive visual information on computer displays via non-visual means, such as through text-to-speech communication. When a screen-reader encounters a image on a webpage, it informs the user that there is an image on the page and reads out the alternative text for that image. Alternative text is a plain text description associated with a graphical entity, that describes the relevant information that that entity

contains. For an infographic to be easily understood by a screen-reader user there needs to be a description of that graphic somewhere, either in the surrounding text on the webpage or in the graphic's alternative text [15]. Unfortunately, however, these descriptions are missing on many webpages and these absences make it difficult for screen-reader users to understand and/or verify how infographics support authors' claims [6,17].

A line chart is a type of infographic that is able to convey information about trends in time series data to readers. This paper presents a method for classifying trends shown in line charts to help address situations where alternative text has not been manually provided for line chart images. While there has been some work towards developing systems that will automatically describe line charts, these have had difficulty generalizing to real-world data [2] and have had difficulty with overfitting training data [8].

In this project we explored an alternative approach to trend classification, with the goal of improving the performance of a template-based description system introduced in Kim et al. 2020 [9]. The system from Kim et al. 2020 [9] separately classifies the line chart trend, predicts axis values, and predicts the chart title for each line chart image. It then uses these predictions to fill in a summary template that describes the chart trend, axis values, and title. We aimed to improved the accuracy of the trend classification step. We utilized the line chart dataset used by Kim and McCoy 2018 [8] and Kim et al. 2020 [9] with some modifications made to address ambiguities in between labels. We modeled our approach to image classification after natural language techniques, taking inspiration from bag-of-words (BoW) approaches for computer vision, which model image classification tasks in a way that is analogous to text-classification tasks. We incorporated distributional semantics into the way we process these visual words, learning distributive representations of the visual words via the Word2Vec architecture that we encoded contextual similarity between words in the word embedding space. We then trained four linear SVM classifiers, two simple baseline classifiers trained on BoW inputs and extracted visual features, and two using the learned distributive representations of the visual words. Comparing the performance of these models with the state-of-the-art CNN architectures, We found that the SVM classifiers trained on the learned word embeddings outperformed the other two baseline classifiers, and achieved slightly better accuracy than a state-of-the-art CNN architecture.

Our contributions are:

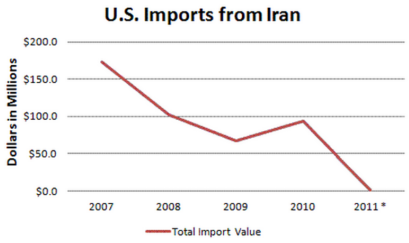
- A method of extracting visual words from line chart images;
- A method of learning embedding of line chart visual words, by using the Word2Vec architecture; and
- A linear SVM classifier that performs comparably to the state-of-the-art CNN on two-class trend classification.

2 Dataset

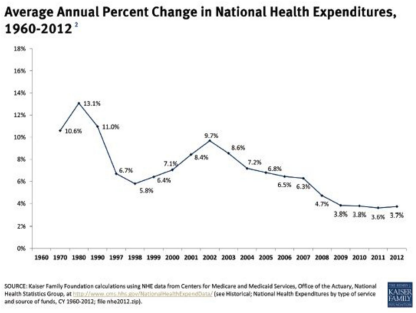
We used the dataset from a previous line chart description generation project, Kim and McCoy 2018 [8]. This dataset contains 998 line-chart images, split into a training set of 848 images and a test set of 150 images. The images are labeled by their trend, with a total of six possible class labels: rising, falling, stable, changing, big-jump, or big-fall.

We noticed several challenges with this dataset. There was significant class imbalance, as there were very few examples stable, big-jump, and big-fall graphs, less than forty for each. As a result, it unlikely that a model could easily learn relevant features for classifying members of these classes. Furthermore, it is difficult to evaluate the quality of the classifier presented in Kim and McCoy 2018 [8] as the paper only presents an accuracy score for the classifier, which is not an ideal performance metric for a dataset with significant class imbalance

Another potential issue we noticed was ambiguities between classes. For example, it seemed reasonable to us that a line chart could show both a changing trend and a trend with a big-fall, or a trend that was rising with big-jump. Upon further inspection it seemed like the differences between the big-jump and big-fall classes and other classes was fairly difficult to grasp. An example of this is shown in Figure 1. In this case it is unclear to why one image is labeled as big-fall trend while the other is labeled as a falling trend.



(a) Example of a big-fall trend



(b) Example of a falling trend

Fig. 1: Demonstration of the ambiguity between images classified as big-fall vs falling as both images show similar trends despite being given different class labels.

Because of these issues we chose to modify dataset for binary classification as opposed to six-class classification, classifying instances as either rising or not-rising, thus reducing ambiguity between classes. The version of the dataset contains 843 training examples and 150 testing examples. The class distribution is shown in Figure 2.

Dataset	Rising	Not-Rising
Train	314	529
Test	47	103

Fig. 2: Distribution of class labels across the modified two-class Kim and McCoy 2018 [8] training and testing datasets

3 Related Work

3.1 Information Graphic Description Generation

There are a few papers that provide approaches to automatic line chart captioning. Some of these approaches simply label these images as line charts or extract chart titles [1,13]. Others provide more detailed descriptions of the line chart images [2,8]. Early approaches to generating detailed line chart description classified chart trends by performing time series segmentation on data points in the charts, then used the resultant segments and other handcrafted features to label the charts via a Bayesian network inference model [18]. This approach required inputs in the form of line chart data points, making it limited in its use, as these data points had to be extracted from the line charts manually in most cases [11]. Later systems were able to perform six-class classification of chart trends with high accuracy (around 69%) via CNN architectures [8]. Unfortunately, 69% accuracy is still not high enough for screen-reader users to rely on summaries that use these model for trend classification.

A major challenge for automatic line chart summarization is that it is difficult to gather annotated line chart images. As a result, datasets are small and more complicated CNN-based models like the one presented in Kim and McCoy 2018 [8] have a tendency to overfit training data. The system introduced in this paper attempts to address the difficulties associated with the limited availability of annotated data by utilizing NLP inspired self-supervised techniques to learn a smaller input space with unlabeled images that we can then use to train a classifier on labeled images.

3.2 Prototype Learning

Recognizing that trend classification is a crucial part of automatic line chart description generation systems and that current state-of-the-art systems have room to improve in this area, we began to look for different approaches that might yield better results. One paper that we found potentially useful was Chen et al. 2019 [3], which provided an explainable approach to image classification with high accuracy. In this work, classifications were performed by comparing image patches from input images against image patches that represented each class. While this approach was able to provide reasonable explanations for classifying photographic images, it lead to underwhelming results on our dataset. Our evaluation of these results led to two important insights about how our

dataset differed from the datasets explored in Chen et al. 2019 [3]: (1) compared to photographic images, line charts contain a relatively little variation in visual content, they are mostly made of line segments and text and (2) components within a line chart have semantic meanings and there are clear semantic and syntactic relationships between components within line chart images, e.g. almost all upward diagonal line segments are semantically similar in that they all indicate a rising section of the chart, and most diagonal line segments of any orientation serve similar syntactic functions as diagonal segments are generally found in the trend of a line chart, rather than in the title or on the axes of the charts. This led us to wonder if it might be useful to treat patches in line chart images analogously to how we treat words in text.

3.3 Bag of Words for Computer Vision

With this question, we looked to the *Bag of Words* (BoW) approach for computer vision. BoW approaches for computer vision represent images in a way that is reminiscent of bag-of-words document representations found in NLP. BoW approaches for computer vision represent images as histograms of visual words [16]. In Csurka et al. 2004 [4] images "visual words" were vector descriptors for features automatically detected within the image that were then mapped to a "vocabulary", learned by clustering the feature descriptors to map them into bins. Each bin was treated as a visual word. In this project, we follow a similar approach, learning a vocabulary of visual words automatically by performing k-means clustering on feature descriptors. Instead of just using BoW representations for the images, however, we also explore alternative methods of embedding the visual words that occur within an image.

3.4 Distributional Semantics

Many NLP task require some way of modelling word meaning and similarities between words. This is generally accomplished via *vector semantic embeddings* [7]. The motivation behind vector semantic embeddings is called the distributional hypothesis. The hypothesis states that semantic relationships between words can be quantified based on the contexts that the words occur in. In other words, words that occur in similar contexts will likely have similar meanings. Vector semantic embeddings look to model this contextual similarity within a vector space.

An effective way of mapping words to such a vector space is to learn distributed representations for words [7]. In Mikolov et al. 2013 [10] word vector representations were learned via a neural network trained on self-supervised tasks. One of the key benefits of the architecture proposed in Mikolov et al. 2013 [10], generally referred to as Word2Vec, is that this task the networks is being trained on is self-supervised, meaning it requires no manually labeled data. In this paper we utilize Word2Vec architecture to learn word embeddings that model contextual similarity between visual words extracted from line chart images.

4 Architecture and Methodology

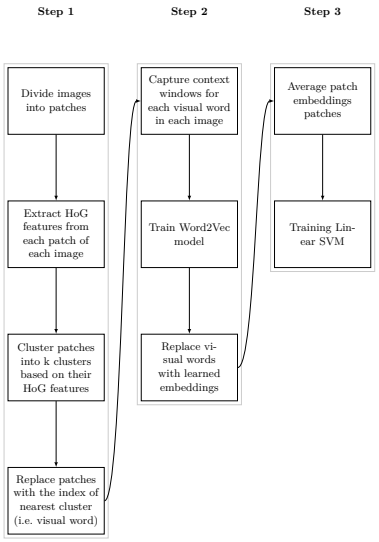
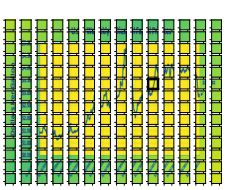


Fig. 3: Training process

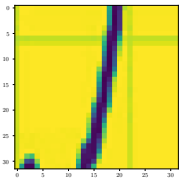
The process of training the classifier can be broken up into 3 major steps: (1) unsupervised clustering to extract words from image patches taken from unlabeled line chart images; (2) self-supervised embedding learning on unlabeled clustered image patches; and (3) classification via a linear support vector machine (SVM) trained on labeled line chart image data, using the learned embeddings to form the input space. This process is shown in Figure 3. Line chart images are then classified by mapping the image patches to visual words, averaging the embeddings of all of the visual words in the image, and then using the SVM classifier to classify the image using thing averaged visual words embeddings as the input to the classifier.

4.1 Forming the Vocabulary

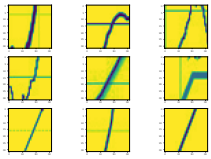
Training images were transformed to be 448×448 pixels with 3 channels. These images were then divided into 196 non-overlapping regions of 32×32 pixels with 3 channels. The vocabulary of visual words was learned from these $32 \times 32 \times 3$ image patches. Using this vocabulary, each patch is then mapped to the most similar cluster to it. An example of this mapping is show in Figure 4. We used Histograms of Oriented Gradients to extract features from patches, and then used k-means clustering to cluster patches together based on those features, forming the vocabulary.



(a) Original image divided into patches



(b) Example patch (bold outlined in sub-figure 4a)



(c) The cluster that the patch shown in sub-figure 4b is assigned to

Fig. 4: Word extraction process for an image

Histogram of Oriented Gradients In order to create a vocabulary from image patches we needed to map images into a space where the distance between visually similar line chart image patches in that space would be small. To do this, we would need a feature descriptor that would:

- lead to similarity between patches containing line segments of similar shape and trajectory but of slightly varying size, varying luminosity, and/or with local changes position and orientation;
- lead to dissimilar vectors for two patches that contain the same line segment except for significant differences in global orientation; and
- produce a fixed size output that could be easily used for clustering.

Given these requirements, we settled on Histogram of Oriented Gradients (HoG) as an acceptable method, as it satisfies the above requirements.

***k*-Means Clustering** With the features computed we then needed to form a vocabulary of visual words. For each patch of each image in the training dataset we computed an HoG feature descriptor. We then used k-means clustering to group the image patches into 500 clusters ($k = 500$). This value was chosen via informal experiments with different values of k. We found that by using 500 clusters we were able to get visually consistent clusters that still had sufficiently large numbers of member patches. We used the cluster centers as the "visual words". Each cluster described a set of visually similar patches. Using these each cluster as word labels, we formed a vocabulary 500 "words". We then replaced each patch in each image with the index of the nearest cluster to the given patch's HoG descriptor.

4.2 Line Chart Embeddings

Once we were able to form the dictionary of visual words and map each patch in each of the training images to a corresponding visual word, we used the resultant matrices of words as documents from which we could learn word embeddings. Mikolov et al. [10] presents two different architectures for Word2Vec: (1) word embeddings are learned via a neural network trained on the continuous-bag-of-words task, where the network predicts a center word from surrounding context words; and (2) word embeddings are learned via a neural network trained on the continuous-skip-grams task, where the network predicts the surrounding context words given a center word [10]. Both of these architectures can be trained in a self-supervised fashion, as center words and the words surrounding them can be automatically obtained from the training corpus. This process requires setting a context window size which determines how many surrounding words are added to a given center word's context. Because my training corpus consisted of two-dimensional matrices of words as opposed to sentences, we had to modify how contextual words were obtained for center words. Instead of just taking the words to the left and right of a given center words, we formed square context windows around the center words. Using 3×3 windows, a given word's context were the 8 words above, below, to the left of, to the right of, and diagonal from it. Using these contextual windows, we were able to learn embedding of the words using both the continuous-bag-of-words and continuous-skip-grams approach, without any additional modifications.

4.3 Classification

We trained 4 Linear SVM classifiers trained on the following inputs:

- Bag-of-word histograms of line chart images (this was a baseline model)
- Concatenated HoG feature descriptors of the image patches with line chart images (this was a baseline model)
- Averaged continuous-bag-of-word Word2Vec word embeddings of line chart images
- Averaged continuous-skip-grams Word2Vec word embeddings of line chart images

We used 5-fold cross validation on the training data to determine the regularization parameter to use for each classifier, from a set of possible values, $\{0.1, 1, 10, 100, 1000\}$.

5 Implementation

To implement our approach, we used on the following Python machine learning libraries:

- `gensim`,

– `sci-kit-learn`, and
– `sci-kit-image`.

For feature extraction we used the `sci-kit-image` library's HoG implementation. For clustering we used the `sci-kit-learn` library's implementation of k means clustering [12]. To learn word embeddings, we used the `gensim` library's implementation of the continuous-bag-of-words Word2Vec model [14]. For the linear SVM classifier, we used the `sci-kit-learn` library's linear SVM classifier, which uses the `liblinear` library, using grid search to select the regularization parameter for the SVM models [5,12].

6 Experiments and Results

6.1 Classification Task

Using the dataset described in Section 2, we evaluated the 4 different linear SVM classifiers trained on: bag-of-word histogram inputs (BoW SVM), concatenated patch HoG vectors (HoG SVM), averaged continuous-bag-of-word Word2Vec embeddings (CBoW SVM), and averaged continuous-skip-grams Word2Vec embeddings (SG SVM). We also compared the accuracies to a ResNet classifier pre-trained on ImageNet (ResNet).

6.2 Results

The results of our experiment are shown in Figure 5. As expected, the models trained with Word2Vec embedding out-performed the other two SVM models in all categories. The SVM model trained on CBoW Word2Vec embeddings performed best out of all of the SVM models, with an accuracy of 91%, a macro-averaged f1-score of 89%, and a weighted f1-score of 91% for the rising class. Comparing its accuracy to that of the ResNet classifier, the SVM trained on CBoW embeddings performed only slightly better that the ResNet classifier's 89% accuracy. The Word2Vec model outperformed the HoG and BoW models reducing the error rate by 8% and 12% respectively, and appeared to perform comparably with the ResNet classier, with an improvement of only 2% in error rates.

Classifier	Accuracy	Macro-Avg. F1	Weighted F1
BoW SVM	79%	77%	79%
HoG SVM	83%	81%	83%
CBoW SVM	**91%**	**89%**	**91%**
SG SVM	85%	83%	86%
ResNet	89%	87%	89%

Fig. 5: Comparison of classifier performance. Best performance in each category is shown in bold.

7 Discussion

The results above show that while our system did not improve performance much over the ResNet classifier, incorporating learned visual word embeddings as inputs to classifiers does seem to be a viable approach for trend classification. This process is likely more computationally efficient and because linear SVM models cannot model complex non-linear relationships, it is less likely to overfit training data. Furthermore, it should not be exceptionally difficult to gather larger amounts of unlabeled images of line charts. This unlabeled data could then be utilized in the vocabulary and embeddings training process to create, as neither requires labeled data.

It must be noted, however, in that this model was evaluated on a two-class classification task; for summary generation application, a trend classifier likely will need to be a multiclass classifier. Future work will require a dataset to be constructed that addresses the problems in Section 2, and a multiclass classifier will need to be evaluated.

8 Conclusion

In this paper we studied a novel approach to trend classification. We proposed a method of extracting visual words from line chart images and used those words to learn semantic word embeddings via the Word2Vec architecture. We observed that it was possible to achieve modest performance gains over CNN-based approaches by using visual word embedding. Future work will look utilize more unlabeled data to learn embeddings and extend this process to multiclass trend classification.

Acknowledgements This material is based upon work supported by the National Science Foundation under Grant No. 1954364.

References

1. Böschen, F., Scherp, A.: Multi-oriented text extraction from information graphics. In: Proceedings of the 2015 ACM Symposium on Document Engineering. p. 35–38. DocEng '15, Association for Computing Machinery, New York, NY, USA (2015). https://doi.org/10.1145/2682571.2797092, https://doi.org/10.1145/2682571.2797092
2. Carberry, S., Elzer Schwartz, S., Mccoy, K., Demir, S., Wu, P., Greenbacker, C., Chester, D., Schwartz, E., Oliver, D., Moraes, P.: Access to multimodal articles for individuals with sight impairments **2**(4) (jan 2013). https://doi.org/10.1145/2395123.2395126, https://doi.org/10.1145/2395123.2395126
3. Chen, C., Li, O., Tao, C., Barnett, A.J., Su, J., Rudin, C.: This Looks like That: Deep Learning for Interpretable Image Recognition. Curran Associates Inc., Red Hook, NY, USA (2019). https://doi.org/10.48550/arXiv.1806.10574, https://dl.acm.org/doi/10.5555/3454287.3455088

4. Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. Work Stat Learn Comput Vision, ECCV **Vol. 1** (01 2004)
5. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A library for large linear classification. Journal of Machine Learning Research **9**, 1871–1874 (2008)
6. Gleason, C., Carrington, P., Cassidy, C., Morris, M.R., Kitani, K.M., Bigham, J.P.: "it's almost like they're trying to hide it": How user-provided image descriptions have failed to make twitter accessible. In: The World Wide Web Conference. p. 549–559. WWW '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3308558.3313605, https://doi.org/10.1145/3308558.3313605
7. Jurafsky, D., Martin, J.: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, vol. 3, chap. 6 (12 2021)
8. Kim, E., McCoy, K.F.: Multimodal deep learning using images and text for information graphic classification. In: Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility. p. 143–148. ASSETS '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3234695.3236357, https://doi.org/10.1145/3234695.3236357
9. Kim, E., Onweller, C., McCoy, K.F.: Information graphic summarization using a collection of multimodal deep neural networks. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 10188–10195. IEEE (2020). https://doi.org/10.1109/ICPR48806.2021.9412146
10. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of the First International Conference on Learning Representations. ICLR (2013). https://doi.org/"10.48550/arXiv.1301.3781", http://arxiv.org/abs/1301.3781
11. Moraes, P.S., Carberry, S., McCoy, K.: Providing access to the high-level content of line graphs from online popular media. In: Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility. W4A '13, Association for Computing Machinery, New York, NY, USA (2013). https://doi.org/10.1145/2461121.2461123, https://doi.org/10.1145/2461121.2461123
12. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
13. Poco, J., Heer, J.: Reverse-engineering visualizations: Recovering visual encodings from chart images **36**(3), 353–363 (jun 2017). https://doi.org/10.1111/cgf.13193, https://doi.org/10.1111/cgf.13193
14. Řehůřek, R., Sojka, P.: Software Framework for Topic Modelling with Large Corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. pp. 45–50. ELRA, Valletta, Malta (May 2010), http://is.muni.cz/publication/884893/en
15. Sharif, A., Chintalapati, S.S., Wobbrock, J.O., Reinecke, K.: Understanding screen-reader users' experiences with online data visualizations. In: The 23rd International ACM SIGACCESS Conference on Computers and Accessibility. ASSETS '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3441852.3471202, https://doi.org/10.1145/3441852.3471202

16. Szeliski, R.: Recognition, pp. 273–331. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-030-34372-9_6, https://doi.org/10.1007/978-3-030-34372-9_6
17. WebAIM: The webaim million (2021), https://webaim.org/projects/million/, https://webaim.org/projects/million/
18. Wu, P.: Recognizing the Intended Message of Line Graphs: Methodology and Applications. Ph.D. thesis, University of Delaware, USA (2012). https://doi.org/10.1007/978-3-642-14600-8_21, aAI3499881