

**#Hashtag**  
**Final Report**  
**Instructor: Jonathon Rose**

**Team members:**  
Tianqi Huang, 1004029712  
Yifang (Evan) Pan, 1004192759

Word count: 1995 words (not counting title page and references)

## Introduction:

This project aims to generate content-relevant hashtags for foodie Instagram posts based on the image in the post. This is useful as hashtags can be used for self-promotions, as well as analyzing market trends on social media, making our project relevant to average social media users and corporate market analysts alike. The scope is limited to foodie posts as this genre of posts does not contain gifs or videos.

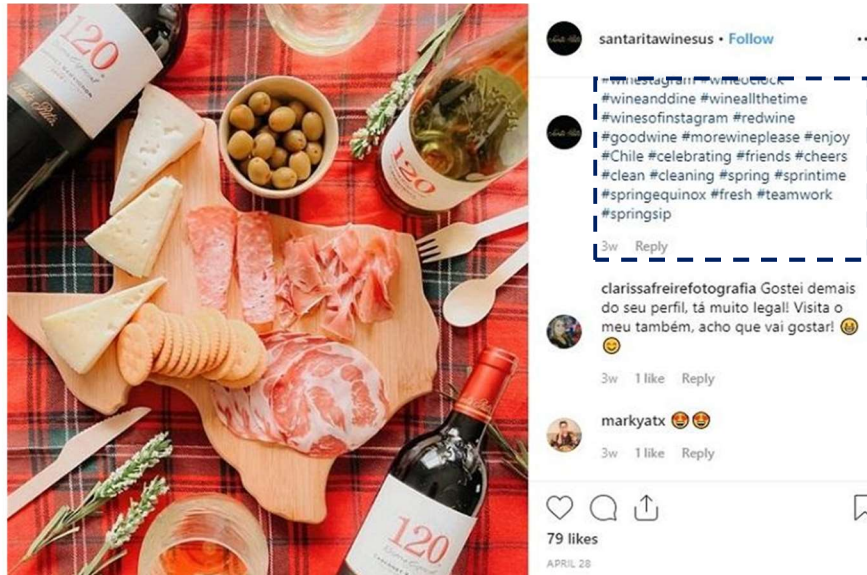


Figure 1. A typical foodie Instagram post contains aesthetically placed food and description of the food displayed. The content (image) of the post appears on the left side of the image, and the hashtags appears as the blue block of text containing phrases started with the pound sign

We have chosen machine learning to be our approach, as image classification is a very complex problem to feature engineer and has historically been more solvable by machine learning. And since both the data (Instagram posts) and labels (hashtags) are readily available online, a relatively large dataset can be easily curated with the use of web crawlers.

## Background and Past work:

Since hashtags are common tags for images, image classification using hashtags have been done in the past. Facebook [1] has experimented using hashtag-labeled images to pre-trained models. For their 2018 paper [1], they have obtained 17 billion images, and approximately 17k hashtag labels (by grouping hashtags with similar meanings). In this model, they have used an image size of 224x224 and trained it on a 101-layered CNN. Their resultant model achieved a top-1 accuracy of 85.4% and a top-5 accuracy of 97.6%.

On Instagrammer's hashtag generator [2] is a similar product to ours. It's able to generate a list of hashtags based on an input image. Their approach seems to be performing a multi-class classification to identify objects in the image, then select hashtags that are related to the object.

## Data and Data Processing:

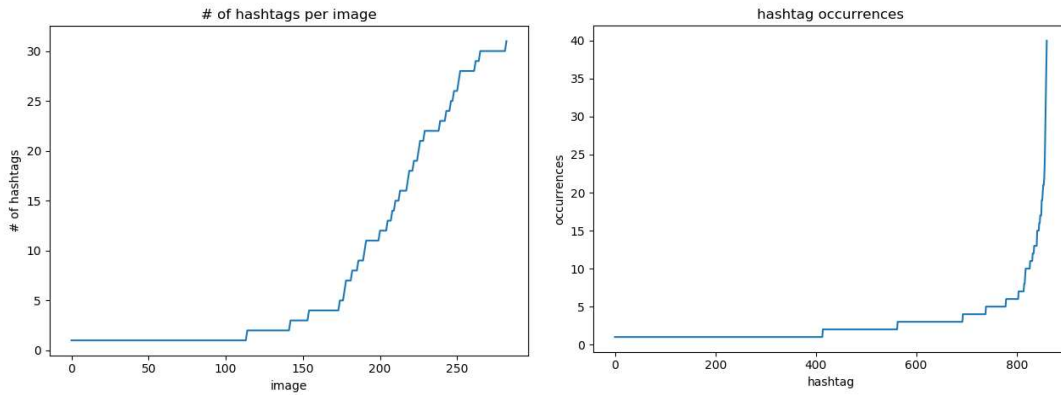
The data is obtained using an Instagram crawler by Huaying Tsai [3] which obtains Instagram posts by the ID of users. A set of 100 food Instagram accounts is randomly selected from many feature articles [4] [5] [6]. The crawler then crawls all the images from the most recent 5 and 100 posts of those accounts for the overfit dataset and large dataset respectively, theoretically collecting 500 and 10000 images respectively. However, due to the updates of Instagram policies and issues with the crawler, the data has the following counts:

Name	Number of images	Number of hashtags
Over-fit	286	562
Large	2576	3048

*Table 1. Count for overfit and large datasets*

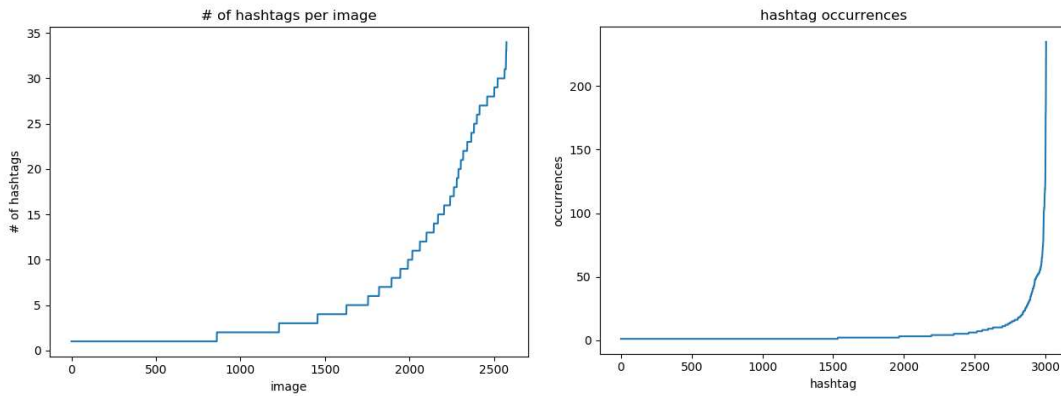
As a dataset for a multilabel classification problem, the dataset has many more hashtags (labels) than samples. The occurrence of each hashtag and number of hashtags for each image varies in a wide range as shown below:

### Overfit Dataset:



*Figure 2. Statistics for overfit dataset, the left side shows the number of hashtags that appears under each image in sorted order, the graph on the right shows the number of occurrence in increasing order*

### Large Dataset:



*Figure 3. Statistics for full dataset, the left side shows the number of hashtags that appears under each image in sorted order, the graph on the right shows the number of occurrence in increasing order*

From the number of hashtags per image, we can see that some images have more hashtags than others. This might pose a challenge to a conventional CNN multi-label classifier, as the one-hot-encoded output would have different magnitudes for different inputs. (I.e. some output would have 2 ones, while others might have 30).

## Preprocessing:

By inspecting the data and the presented statistics, the dataset appears to be imbalanced. In addition, Instagram images come in different sizes and may have filters. Thus, the following pre-processing is applied:

1. Resize every image to 200×200 pixels.
2. Use `torchvision.transform.Colorjitter` to address filters.
3. Each image is normalized across all pixels and 3 channels.

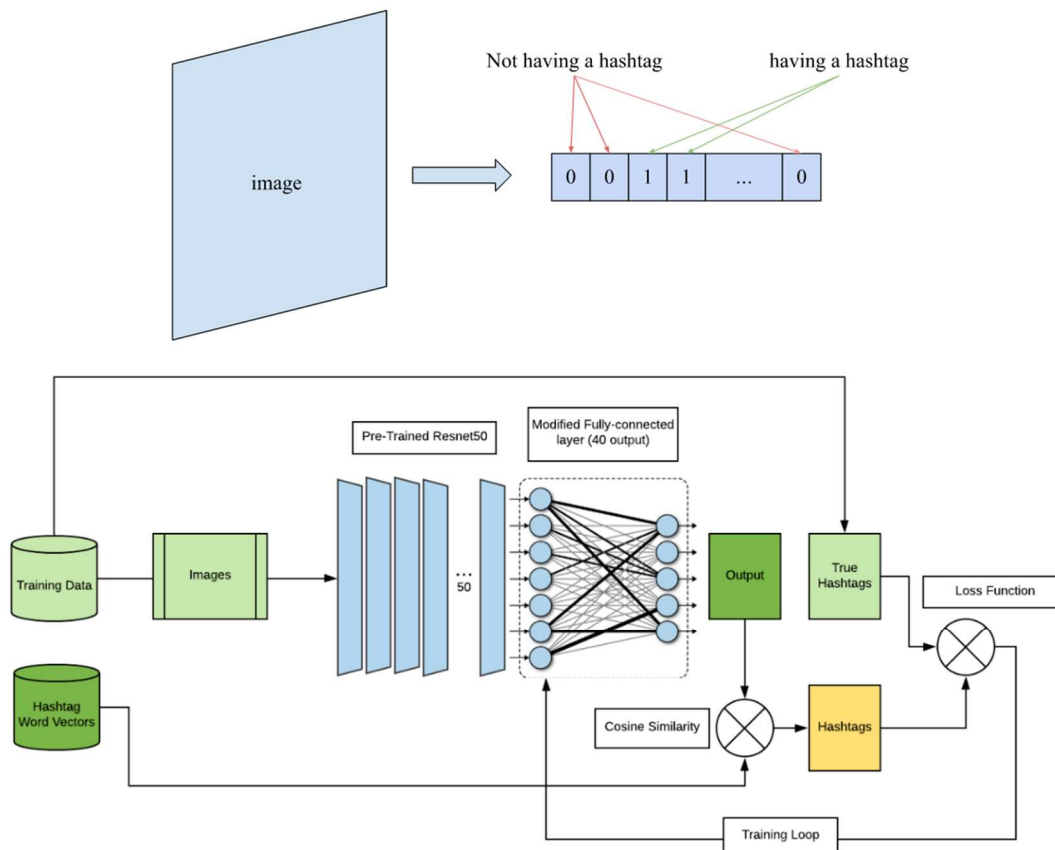
Note that the imbalance is not addressed in preprocessing since the hashtags that appear more often should have more importance. This feature should be reflected in the dataset for the model to learn.

The large dataset was split into train, validation and split as shown in the table below:

Total: 2576	Train	Validation	Test
Ratio	0.7	0.15	0.15
Number of Images	1803	387	386

*Table 2. Train-validation-test split*

## Architecture:



*Figure 4. Overall architecture*

The main idea of our model is to take advantage of the fact that a lot of hashtags have similar meanings (like #SundayMorningBreakfast and #Breakfast). If we can turn the hashtag into embeddings that reflect their closeness, and let the model learn to output these more abstract embeddings instead of the classes, it might be able to reduce the negative effect of having such a massive number of classes.

Based on this idea, the model is designed to have two parts: a word2vec model and a CNN model. The word2vec model takes all hashtags and transforms them into embeddings of dimension 40. The CNN will then produce an output of the same dimension. The cosine distances between the output and each hashtag embedding are then computed to see which hashtags are likely to be correct. A result  $\geq 0.5$  after sigmoid gives a positive prediction (labeled with the hashtag) and the prediction is negative otherwise.

## Hashtag embedding:

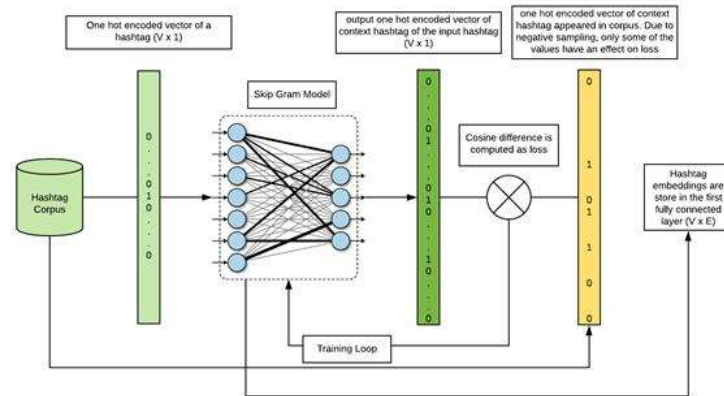


Figure 5. Hashtag embedding

To generate the hashtag embeddings, a skip-gram model with negative sampling found on GitHub is used [7]. We treat a group of hashtags that are all under the same post as a sentence, each hashtag as an individual word, and the hashtags around it as the context. (the analogy is shown in figure P) Then the hashtag embeddings are trained by converting the collected data into a corpus and feeding the corpus into the skip-gram model.

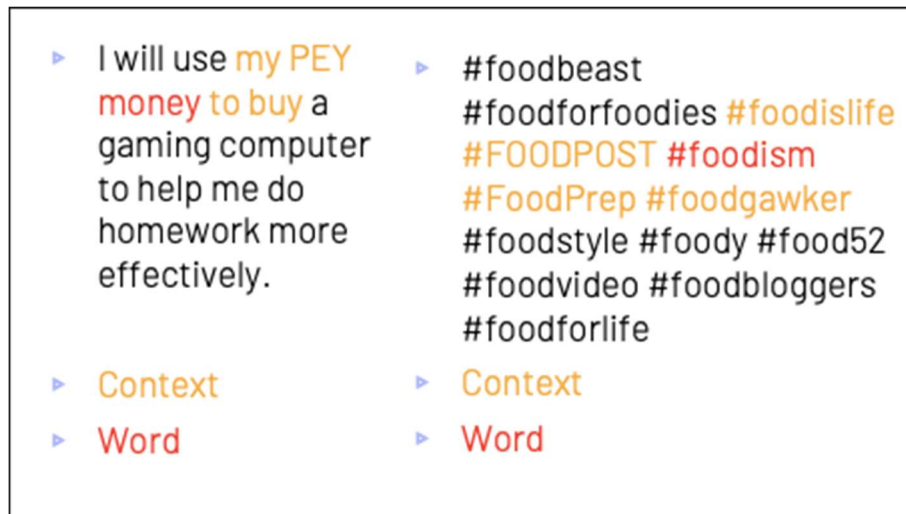


Figure 6: Analogy between a conventional input to word2vec and our model.



```

myfavourite sweettooth
frommydiningtable
decorativegourdeason wildfoodlove
vegan glutenfree sponsored
f52grams feedfeed huffposttaste buzzfeed eeeeeats beautifulcuisines thekitchn foodandwine forkyeah yahoofood onthetable
buzzfeedfood vscofood foodstyling foodphotography lifeandthyme eater heresmyfood food52 gloobyfood foodgawker eattheworld
eatfamou tastintable dailyfoodfeed spoonfeed foodbeast hautecuisines foodblogfeed onmytable
f52grams feedfeed huffposttaste buzzfeed eeeeeats beautifulcuisines thekitchn foodandwine forkyeah yahoofood onthetable
buzzfeedfood vscofood foodstyling foodphotography lifeandthyme eater heresmyfood food52 gloobyfood foodgawker eattheworld
eatfamou tastintable dailyfoodfeed spoonfeed foodbeast hautecuisines foodblogfeed onmytable
theartofescapismcooking ladyandpupscookbook holidaygiftguide f52grams feedfeed huffposttaste buzzfeed eeeeeats
beautifulcuisines thekitchn foodandwine forkyeah yahoofood onthetable buzzfeedfood vscofood foodstyling foodphotography
lifeandthyme eater heresmyfood food52 gloobyfood foodgawker eattheworld eatfamou tastintable dailyfoodfeed spoonfeed
foodbeast
f52grams feedfeed huffposttaste buzzfeed eeeeeats beautifulcuisines thekitchn foodandwine forkyeah yahoofood onthetable
buzzfeedfood vscofood foodstyling foodphotography lifeandthyme eater heresmyfood food52 gloobyfood foodgawker eattheworld
eatfamou tastintable dailyfoodfeed spoonfeed foodbeast hautecuisines foodblogfeed onmytable
f52grams feedfeed huffposttaste buzzfeed eeeeeats beautifulcuisines thekitchn foodandwine forkyeah yahoofood onthetable
buzzfeedfood vscofood foodstyling foodphotography lifeandthyme eater heresmyfood food52 gloobyfood foodgawker eattheworld
eatfamou tastintable dailyfoodfeed spoonfeed foodbeast hautecuisines foodblogfeed onmytable
f52grams feedfeed huffposttaste buzzfeed eeeeeats beautifulcuisines thekitchn foodandwine forkyeah yahoofood onthetable
buzzfeedfood vscofood foodstyling foodphotography lifeandthyme eater heresmyfood food52 gloobyfood foodgawker eattheworld
eatfamou tastintable dailyfoodfeed spoonfeed foodbeast hautecuisines foodblogfeed onmytable
f52grams feedfeed huffposttaste buzzfeed eeeeeats beautifulcuisines thekitchn foodandwine forkyeah yahoofood onthetable
buzzfeedfood vscofood foodstyling foodphotography lifeandthyme eater heresmyfood food52 gloobyfood foodgawker eattheworld
eatfamou tastintable dailyfoodfeed spoonfeed foodbeast hautecuisines foodblogfeed onmytable
nothingfancycookbook

```

Figure 7: Hashtag Corpus without the pound sign, Skip gram will treat each line as sentence, and each space-separated string as a word

The key hyper-parameters to optimize for are the window size, embedding size, and the number of epochs, as the rest of the hyperparameters have default values from the model we found. The window size tells the model how many hashtags should be considered as the context, and the embedding size is the size of the embedding. The chosen hyperparameters chosen are shown below.

Number of Iterations	Embedding size	Window size (+ and -)
10	40	5
Learning rate	negative sampling count	batch size
0.25	5	128

Table 3. hyperparameters of the Skip-gram Model

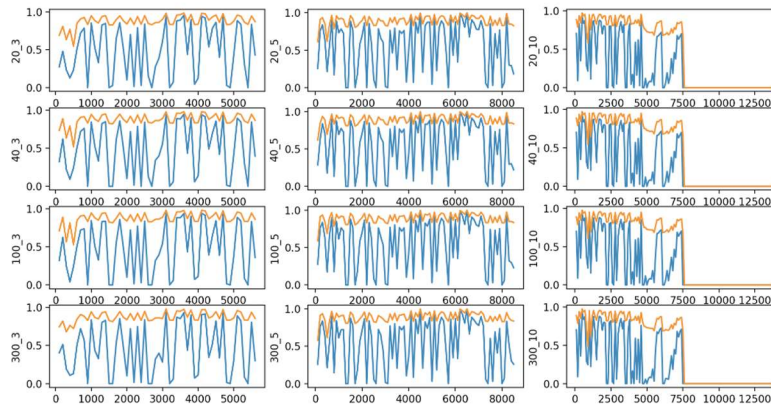


Figure 8: Grid search result for skip gram model. The y-axis is the accuracy (orange) and the f1 score (blue) of the model, and the x-axis is the number of batches. The graphs are labeled by the y-axis label, in the format of <Embedding size> <window size>. All models are trained for 20 epochs.



A grid search is used to find the best embedding size and window size. From the result we can see when the window size gets to 10, the model diverges fairly quickly (observing the hashtag embedding shows all values of the embeddings reached either positive or negative infinity). Further testing shows that the model with a window size of 5 also diverges after 21-23 epochs. For this reason, the window size of 5 and epoch of 10 are chosen. The testing also shows that the embedding size has little effect on accuracy, therefore a small but conservative size of 40 is chosen.

Image processing:

The image part utilizes Resnet50, which is a pre-trained CNN with 50 layers. The output layer is replaced with a 40-neuron layer to output the same dimension as the hashtag embedding to be compared using the cosine similarity as specified before. The training will be done by freezing all the convolution layers, and the following Hyper-parameters are used to train the model.

Loss function	activation function	Learning Rate
KLD and BCEwithLogits	Leaky Relu	0.00001
Epochs	Optimizer	Additional tricks
50	Adam	Batch Norm

*Table 4. Hyperparameters of the transfer learning model*

In addition to this model that utilizes Resnet50 as the CNN, a simpler model that utilizes the CNN from assignment 4 is used as a control to see the effect of pretraining. It is trained with the same set of hyperparameters as Resnet50.

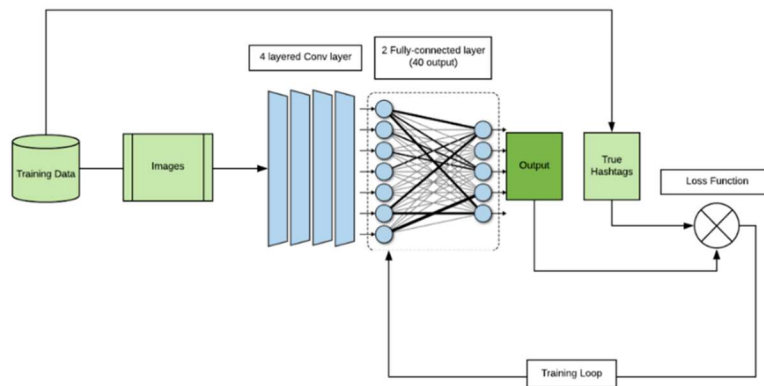
Loss function:

In our attempt to overfit the model, it is discovered that the model quickly learned to output all zeros to fool the loss when using MSE or Cross Entropy. Therefore, we've resolved to use a linear combination of loss functions to ensure the model doesn't output all zeros.

The loss function used is a combination of Kullback-Leibler Divergence (KLDiv) loss and Binary Cross-Entropy (BCEwithLogits) Loss. The KLDiv loss promotes the number of positive prediction while the BCEwithLogits promotes the overall accuracy (which tends to push the model to make all negative predictions). Using a linear combination of both helps maximize the F2 score. The final loss function is determined to be:

$$Loss = KLDiv + 0.1 \cdot BCEwithLogits$$

### Baseline Model:



*Figure 7. Baseline model diagram*

The baseline model consists of a CNN much like assignment 4. The CNN has 4 convolutional layers with a max-pooling layer following each. The kernel sizes are all 3x3. All max-pooling layers have size 2 and stride 2. The output will simply be the one-hot-encoded hashtags. An element  $\geq 0.5$  gives a positive prediction and it gives a negative prediction otherwise. The following hyperparameters are used.

Loss function	activation function	Learning Rate	Kernel per layer
KLD and BCEwithLogits	Leaky Relu	0.00001	10
Epochs	Optimizer	Additional tricks	Kernel size
50	Adam	Batch Norm	3x3

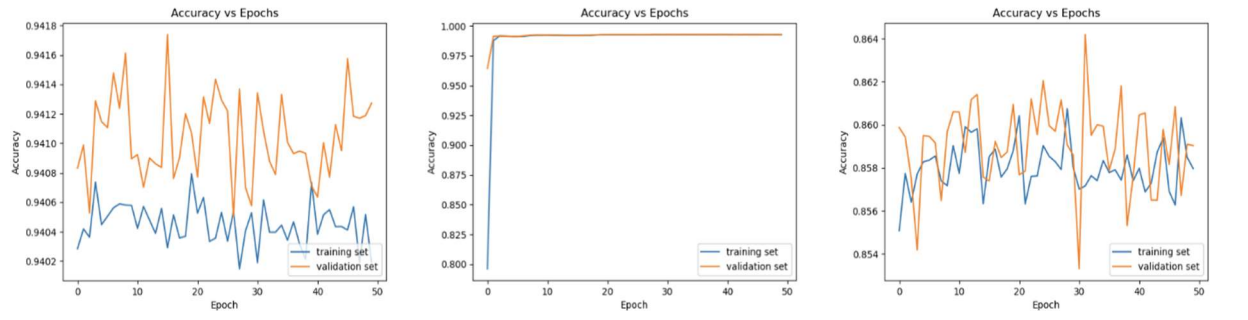
*Table 5. Hyperparameters of the Baseline model*

## Results and Discussion:

For a multi-class multi-label classification problem with about 3000 classes such as our problem, accuracy on its own is not necessarily a very reliable measure of performance as the model can simply output all zero to full the accuracy. In our case, in addition to accuracy, the F2 score (weighted average of both precision and recall, with a stronger emphasis on precision) is also used. The F2 score is calculated with the following equation:

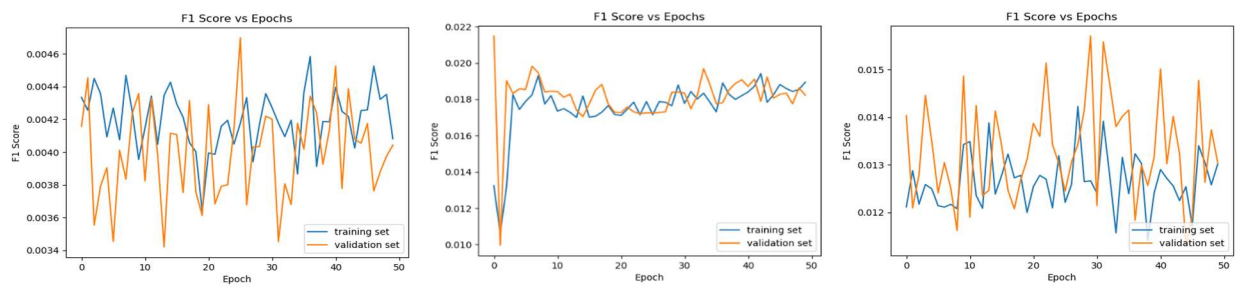
$$F2 \text{ score} = \frac{5 \cdot \text{precision} \cdot \text{recall}}{4 \cdot \text{precision} + \text{recall}}$$

### Quantitative:



*Figure 8: Training and validation accuracy of the baseline (left), CNN with embedding (middle) and Resnet with embedding (right) on the overfit dataset*

From the training curves, it is surprising to see that almost all the models have achieved an accuracy of around 90%. However, accuracy can be cheated simply by outputting all zeros. The F2 score curve of the models is shown below.



*Figure 9: Training and validation accuracy of the baseline (left), CNN with embedding (middle) and Resnet with embedding (right) on the full dataset*

The f2 score trends show that for all models, the learning stopped at very early, then the f2 score oscillates around the same value. Which indicates that the model is not doing much learning.



Accuracy	Baseline	CNN-with embedding	Resnet50 with embedding
Train	0.94	0.98	0.861
Validation	0.94	0.98	0.86
Test	0.93	0.97	0.865

F2Score	Baseline	CNN-with embedding	Resnet50 with embedding
Train	0.0446	0.02	0.14
Validation	0.0446	0.02	0.155
Test	0.042	0.019	0.14

*Table 6. Accuracy and F2 score of the three models on the train, validation and test datasets.*

From table 6, we can see that for the F2 score, the models that utilize word-embeddings have achieved a better result. This can be due to the fact that word-embeddings are able to create some level of abstraction for hashtags that generalized some hashtags to effectively reduce the number of classes the model needs to label. One thing that's surprising is that the CNN trained from scratch performs better than the pre-trained model. This might be due to the fact that Resnet is not trained on images of food. In future iterations, we might train the model without freezing the convolution layers.

Qualitative:

Image	Hashtags (top 5)	Hashtags (true)
	#nothingfancycookbook #InstagramWife #Foodintheair #TheKitchen #vineboxpartner	# nothingfancycookbook
	#feedfeed #huffposttaste #beautifulcuisines #foodandwine #foodphotography	#feedfeed #huffposttaste #buzzfeast #beautifulcuisines #thekitchn #foodandwine #forkyeah #yahoofood #onthetable",

The result shows the model is able to predict some related hashtags and we can reason how those hashtags are determined based on the images. The model is successful to a certain degree understanding the common meaning between words and images. In the first example, the model recognized the women and the kitchen in the image, which is exclusively for this sample. It also correctly predicted the hashtag. In the second example, the model mistakenly thought the meat was toast while in fact, the meat does look like bread here. The model also correctly predicts a lot of the hashtags. Though the model may not get a lot of correct hashtags it does produce logically understandable results. F2 score and accuracy may not reflect the true effectiveness of the model. The model demonstrates some ability to interpret from the image and find relatable hashtags.

## Further Discussion:

Besides the issue with the ambiguity of hashtags, another underlying issue is the generation of hashtag embeddings. In our approach, we used word2vec to generate embeddings for hashtags by treating hashtags as words. However, hashtags and words are very different. First of all, hashtags have a much smaller corpus compared to words, there is simply not enough context to learn from. Secondly, a sequence of hashtags is not regulated by grammar and can appear in any order, therefore word2vec models that rely on predicting context around words would not work very well.

Finally, by examining our dataset, we found that hashtags do not always correlate with the content of the image. In the two Instagram posts shown below. Though both contain images of latte art, their hashtags are drastically different. Therefore in future iterations, we might also take location and text description as inputs.



*Figure 10: 2 Instagram posts of latte art. The first one is posted with hashtags mainly associated with the geographical location Brighton, UK, while the second post contains hashtags that are mostly associated with the theme explore.*

## **Ethical Framework:**

One group of stakeholders for our project are Instagram users we collected the data from, which we did without their permission. This action does not respect the autonomy of those users as we are taking advantage of their content without reciprocating. It might also be maleficent if the users do not wish others to see their data, for example for refugees on political asylum.

A second important group of stakeholders would be the end-users of our model, who uses our model to generate hashtags for their posts. This can potentially be very beneficent for those who are new to social media and wish to spread their message to more people, as they can use a hashtag to reach a wider audience, such as political activists and event promoters. However, if this is used by a large group of interest, it might introduce injustice. If large corporations wish to drown out the voice of their competitors, they can flood the feed with posts carrying the same hashtags, preventing the message of their competitors to be seen. In terms of nonmaleficence, it would depend also depend on who uses it, an authoritarian government can use our program to general tags for otherwise untagged posts and perform censoring.



## References

- [1]D. Mahajan et al., "Exploring the Limits of Weakly Supervised Pretraining", 2019. Available: <https://arxiv.org/pdf/1805.00932.pdf>. [Accessed 3 December 2019].
- [2]"Ingramer — x2 More Effective Instagram Bot", *Ingramer.com*, 2019. [Online]. Available: <https://ingramer.com/>. [Accessed: 03- Dec- 2019].
- [3]"huaying/instagram-crawler", *GitHub*, 2019. [Online]. Available: <https://github.com/huaying/instagram-crawler>. [Accessed: 03- Dec- 2019].
- [4]"100 Incredibly Tasty Instagram Accounts for Foodies to Follow", *Photodoto.com*, 2019. [Online]. Available: <https://photodoto.com/instagram-accounts-for-foodies/>. [Accessed: 03- Dec- 2019].
- [5]"<https://www.foodandwine.com>", *Food & Wine*, 2019. [Online]. Available: <https://www.foodandwine.com/news/best-food-instagram-accounts>. [Accessed: 03- Dec- 2019].
- [6]"The 30 Best Instagram Food Accounts You Should Be Following", *BuzzFeed*, 2019. [Online]. Available: <https://www.buzzfeed.com/jesseszewczyk/best-food-instagram-accounts>. [Accessed: 03- Dec- 2019].
- [7]"PengFoo/word2vec-pytorch", *GitHub*, 2019. [Online]. Available: <https://github.com/PengFoo/word2vec-pytorch>. [Accessed: 03- Dec- 2019].

- permission to post video: yes
- permission to post final report: yes
- permission to post source code: yes