# CountCLIP - [Re] Teaching CLIP to Count to Ten

**Anonymous authors**
**Paper under double-blind review**

## Abstract

Large vision-language models (VLMs) are shown to learn rich joint image-text representations enabling high performances in relevant downstream tasks. However, they fail to showcase their quantitative understanding of objects, and they lack good counting-aware representation. This paper conducts a reproducibility study of 'Teaching CLIP to Count to Ten' (Paiss et al., 2023), which presents a method to finetune a CLIP model (Radford et al., 2021) to improve zero-shot counting accuracy in an image while maintaining the performance for zero-shot classification by introducing a counting-contrastive loss term. We contribute to the existing methods by improving the model's performance on a smaller subset of their training data with lower computational resources. We verify these claims by reproducing their study with our own open-source code. The implementation can be found at `https://anonymous.4open.science/r/CountCLIP-FA07`.

## 1 Introduction

In recent years, the development of large Vision-Language Models (VLMs) has significantly propelled the field of representation learning in computer vision, with models like CLIP (Radford et al., 2021) and BASIC (Pham et al., 2023) showcasing their ability to learn robust joint image-text representations. Tasks such as zero-shot classification, segmentation, image captioning, and text-to-image generation benefit from the compositional understanding capabilities of such models. However, these models struggle on counting tasks such as matching an image representation based on the number of the specified objects present, to the text representation of the count corresponding to the image's caption. The popularity of text-to-image and text-to-video models such as Stable Diffusion Video (Blattmann et al., 2023), Sora (Brooks et al., 2024), Lumiere (Bar-Tal et al., 2024), has risen and such models rely heavily on CLIP for their image-text representations. Count-aware models like (Paiss et al., 2023) shall greatly enhance the abilities of these existing architectures, enabling models to produce highly accurate videos with the correct number of entities.

Paiss et al. (2023) suggested shifting the training objective to discriminate between the correct and the incorrect captions associated with the object counts given an image by introducing a counting loss term $L_{count}$ to the VLM's loss function. The paper reported a pipeline to create training data consisting of counting images. A *counting image* is an image whose caption accurately reflects the number of entities in the image, as shown in Figure 1(b). This data is then used to fine-tune a pre-trained VLM by contrasting the representation of the correct caption with that of the synthetically generated counterfactual caption, where the count in the correct caption is randomly swapped to an incorrect one. This way, the VLM is trained to align images with its true count captions and discriminate against the incorrect ones. The trained model is evaluated on the *CountBench* benchmark, an object counting benchmark introduced by the paper. Noting its importance to the contributions to VLM's, this paper seeks to reproduce and build upon their work.

This paper primarily aims at reproducibility while also advancing the existing research. Our keys contributions include:

1. We improved the model's performance on a smaller subset, 640 times smaller than that of the paper's training data and with lower computational resources, beating the baseline by 1.38%.

2. We created and released our own counting image training dataset, making it publicly accessible.

3. We created and made public a more comprehensive version of the CountBench benchmark by manually including images from non-functional URLs.

## 2 Materials and Methods

### 2.1 Dataset

The LAION-400M (Schuhmann et al., 2021) dataset is used to obtain the counting image dataset. Of the 400 million images, they obtained roughly 200,000 counting images. Due to our size and computational constraints, we passed over 2 million images and obtained ∼2,000 counting images. We have made our code and counting set public. As described in the paper, we looked for the sentences that contained the numbers *"two"* to *"ten"* in words. Images were passed through the YOLOv8 (Jocher et al., 2023) object detector, and the count of the most frequent entity was checked with the count in the caption. If they are equal, the image-text pair is added to the counting set, as seen in Figure 1(a) .

The method to generate the counting set was highly unfeasible when running over the full dataset. The training data could have been made public with the URLs being public. The large number of non-functioning URLs in the original dataset was a major bottleneck. At least 30 out of 540 images were unavailable in the *CountBench* due to the URLs of the images being defunct. We contacted the authors of the papers, and they stated that while they possess the image files, it was against their company policy to share raw images sourced from publicly available data. The benchmark is the only one of its kind, and it was carefully curated, to ensure class balance. As a result, any future work measuring counting accuracy must use this benchmark in addition to other methods for fairer and more reproducible comparisons.

### 2.2 Training the model

The authors in the original paper utilized a dual loss function, combining the regular contrastive loss of CLIP ($L_{CLIP}$) with a counting-designated loss ($L_{count}$), weighted by the hyperparameter $\lambda$ (Equation 1).

$$L = L_{CLIP} + \lambda L_{count} \tag{1}$$

The fine-tuning process involved training the model on two different training sets: (i) LAION-400M (Schuhmann et al., 2021), an extensive dataset collected from the web comprising general images and captions, and (ii) a filtered numbered training set $C$, as described in Section 2.1, containing samples with object counts spelled out in the captions. While the regular contrastive loss $L_{CLIP}$ was calculated on all samples, the counting loss $L_{count}$ was specifically computed for samples from set $C$, with $N$ counting image-text pairs in a batch. For each image-text pair $(i_k, t_k)$ in set $C$, a counterfactual caption $t_k^{CF}$ was automatically generated by replacing the number in the original caption $t_k$ with a different random number (e.g., for an image consisting of four parrots, the corresponding true caption "four parrots" can be counterfactualized with "seven parrots").

During each step of the training process, the tuples $(i_k, t_k, t_k^{CF})_{k=1}^N$ are fed into CLIP's text and image encoders, resulting in the generation of their respective embeddings $(ei_k, et_k, et_k^{CF})_{k=1}^N$. The contrastive loss $L_{count}$ was then computed to ensure a high similarity score between the image and the original caption and a low similarity score with the counterfactual caption (Equation 2). This loss function encouraged the model to learn the relationship between the spelled-out number in the caption and the number of objects it referred to. This is shown in Figure 1.

$$L_{count} = -\frac{1}{N} \sum_{k=1}^N \log \frac{\exp(ei_k \cdot et_k)}{\exp(ei_k \cdot et_k) + \exp(ei_k \cdot et_k^{CF})} \tag{2}$$

Furthermore, negative samples were exclusively used in the counting objective $L_{count}$, rather than being added to the batch for the existing contrastive loss $L_{CLIP}$. This selective approach was adopted to better weigh the impact of negative samples on the counting objective. To reproduce the experimental setup
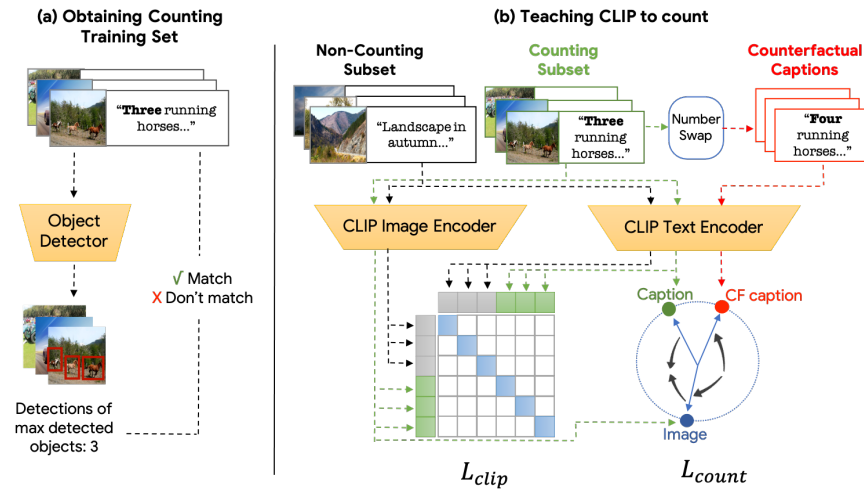
Figure 1: Figure copyright Paiss et al. (2023). (a) Training setup (b) Examples of counting images (c) Examples of noncounting images

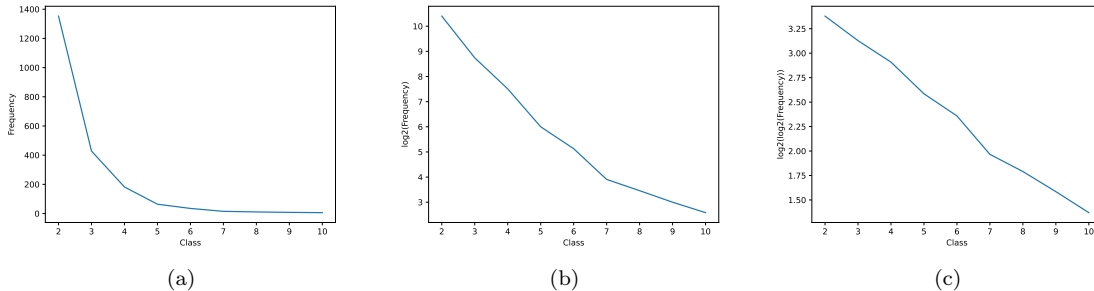Figure 2: (a) Class frequency (b) $log_2(frequency)$ (c) $log_2(log_2(frequency))$

accurately, we followed the described procedures, including creating counterfactual captions and the training regime on the specified datasets.

## 2.3 Balancing lambda

In section 2.2, the hyperparameter $\lambda$ was introduced to adjust the weightage of $L_{CLIP}$ and $L_{count}$ in the final loss function. In the paper, the optimal lambda was reported to be 1.

We introduce a new scheme for setting the hyperparameter $\lambda$ by balancing it on the frequencies of the classes. The main motivation behind this is the severe imbalance in the counting training data for both our counting set as well as Paiss et al. (2023)'s. The paper used a form of undersampling, as they had very a large amount of data avaliable to them. This scheme of choosing $\lambda$ ensures that more focus is given to the less frequent class. This technique allows us to use this modified loss on smaller datasets like ours.

We present three ways to balance lambda, $\lambda_{norm}, \lambda_{modal}, \lambda_{log}$. The $\lambda_{norm}$ in equation (3) setting the focus in proportion to the frequency of classes. The lambda is normalised as it is always a fraction of $\lambda_0$.

$$\lambda_{norm}(class) = (1 - \frac{n_{class}}{n_{total}})\lambda_0 \tag{3}$$

The $\lambda_{modal}$ (given by Equation (4)) sets more focus on less frequent classes, while ensuring that the most frequent (modal) class has the minimum value $\lambda_0$.

$$\lambda_{modal}(class) = \frac{n_{modal}}{n_{class}}\lambda_0 \tag{4}$$

The $\lambda_{log}$ in Equation (5) and $\sigma$ in Equation (6) are computed for all classes. $\sigma_{min}, \sigma_{max}$ are the minimum and maximum values of $\sigma$ after computing it for each class respectively. The training dataset had a class imbalance resembling an exponential curve. By taking the logarithmic transformation as in Equation (6), we can apply a linear scaling to the $\lambda$, while avoiding overfitting on the least frequent classes. The class distributions are plotted in figure 2(a), and resembles an exponential curve. On taking the logarithm, we see that the distribution is now close to linear. We apply the logarithm again to make the distribution as linear as possible for our training data. This is seen in the figure 2. We have tested the model by taking the logarithm once (i.e., $log_2(\frac{n_{total}}{n_{class}})$) for computing $\sigma$, however, it yielded poorer results, so we proceeded with the transformation as per Equation 6.

$$\lambda_{log}(class) = (\frac{\sigma_{class} - \sigma_{min}}{\sigma_{max}} + 1)\lambda_0 \tag{5}$$

$$\sigma(class) = log_2(log_2(\frac{n_{total}}{n_{class}})) \tag{6}$$

$n_{class}$ is the number of examples of the given class, and $n_{total}$ is the total number of counting images in the training dataset. The base lambda $\lambda_0$ is set to 1. During train time, we pass the counting caption to a function to get its appropriate value of $\lambda_{balanced}$ based on its count.

### 2.4 CountPlus

The loss function introduced in the paper (Equation 2) contrasted the correct caption with the counterfactual caption, where the count in the correct caption is randomly swapped to an incorrect one. We experiment by changing the loss so that it contrasts the correct caption with all possible counterfactual captions, where the number is swapped with all possible incorrect values.

$$L_{count+} = -\frac{1}{N} \sum_{k=1}^{N} \log \frac{\exp(ei_k \cdot et_k)}{\exp(ei_k \cdot et_k) + \sum_{j=2; j \neq count}^{10} \exp(ei_k \cdot et_k^{CF})} \tag{7}$$

## 3 Evaluation

The paper also published a new image-text counting benchmark *CountBench* for evaluating a model's understanding of object counting, which we used to benchmark our model. The data is carefully curated and class-balanced, with 540 images. We create embeddings of all the possible combinations for the captions of the images, take the embeddings of the 9 captions and images, and compute the similarity score with each caption. The caption with the highest similarity score is the predicted class. We have normalised the embeddings before the dot product in all the results reported in Table 1. Figure 4 and Figure 5 gives the confusion matrices for the tested configurations.

## 4 Results

The baseline accuracy for a CLIP B/32 model on the CountBench dataset is 27.5% with no training. The confusion matrices for the main comparisons are given in Figure 3. Please refer to Figure 4 and Figure 5 of Appendix A for the confusion matrices for the remaining experiments. The accuracies are reported in Table 1. Paiss et al. (2023) reported the results on both CLIP and BASIC. However, in this paper, we only focus on training CLIP. We have used a learning rate of 5e−6, and 20,000 steps (10 epochs), and a linear warmup in the first half of the steps with a cosine scheduler in the other half. The difference is in the amount of data used to train the model, $b_{size}$ is the batch size, $p$ is the proportion of counting images in a batch, and $n_{count}$ is the total number of counting images in the training data. Paiss et al. (2023): $b_{size} = 32,768, p = 1/32, n_{count} = 200,000$; Ours: $b_{size} = 5, p = 1/5, n_{count} = 2000$.

| Configuration | Accuracy (end of $10^{th}$ epoch) (%) | Max Acc. (with early stopping) (%) |
|---|---|---|
| $\lambda = 1$ with no scheduler | 21.81 | 25.15 |
| scheduler and $\lambda = 1$ (base model) | 22.59 | 25.93 |
| $\lambda_{norm}$ | 21.61 | 25.74 |
| scheduler and $\lambda_{norm}$ | 25.54 | 25.54 |
| scheduler and $\lambda_{norm}$ and $L_{count+}$ | **26.92** | 26.92 |
| $\lambda_{modal}$ | 20.83 | 26.13 |
| scheduler and $\lambda_{modal}$ | 19.84 | 26.33 |
| scheduler and $\lambda_{modal}$ and $L_{count+}$ | 26.33 | **28.88** |
| $\lambda_{log}$ | 19.25 | 24.56 |
| scheduler and $\lambda_{log}$ | 21.81 | 27.90 |
| scheduler and $\lambda_{log}$ and $L_{count+}$ | 22.59 | 27.50 |

Table 1: Zero-shot counting accuracy for the configurations for a CLIP-B/32 model
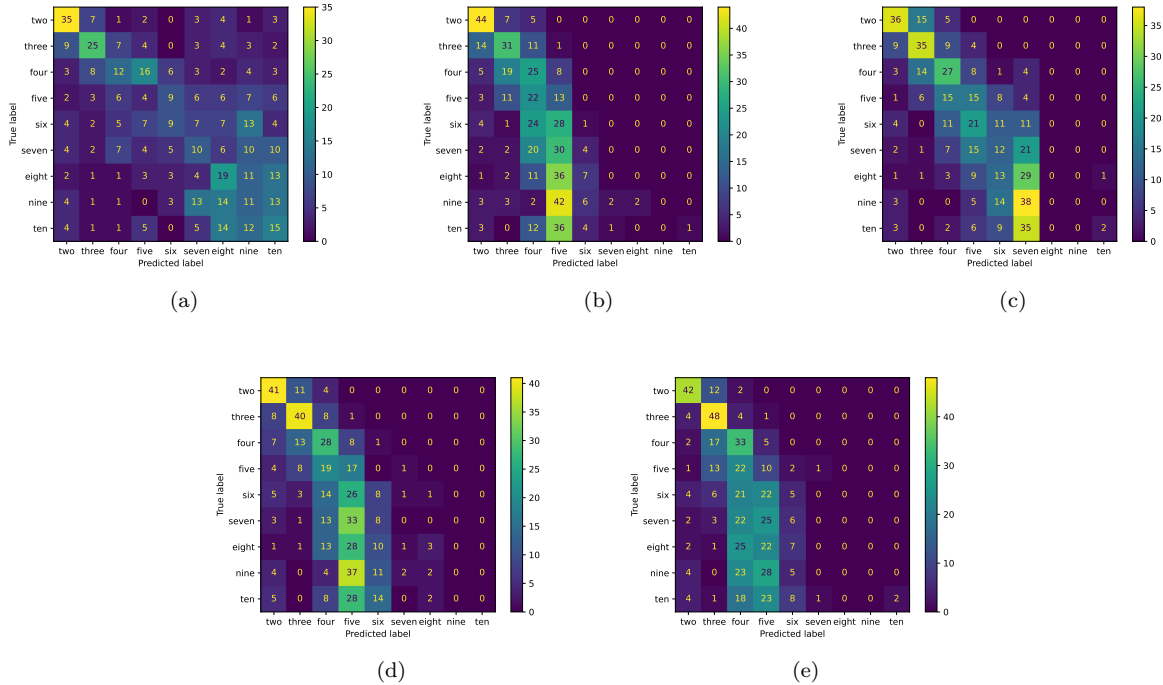
Figure 3: Confusion matrices with early stopping for: (a)baseline (b) scheduler and $\lambda = 1$ (base model) (c) scheduler and $\lambda_{modal}$ and $L_{count+}$ (d) scheduler and $\lambda_{norm}$ and $L_{count+}$ (e) scheduler and $\lambda_{log}$ and $L_{count+}$

## 5    Discussion

We found adapting the auxiliary loss weight as a class weight with an appropriate scheme is an effective way to boost the performance of the model in scenarios with extreme class imbalance and low training dataset sizes. We have also found that changing the counting objective to a multiclass classification loss leads to better performance when used with a scheme for balancing lambda, even exceeding or matching the baseline by upto 1.38% despite the severe lack of data. We performed an ablation of the learning rate scheduler and observed a gain in both maximum accuracy (using early stopping) and the end of 10th epoch in most cases.

Better schemes can be devised to improve accuracy, a limitation of using such schemes is that, while it may increase the overall accuracy, it may also weaken the accuracy of classes that are more data rich. The confusion matrices in Figure 3 show that the models rarely predict the higher-numbered classes, which may be due to the models not learning to classify those classes due to a lack of training data available for those classes (refer to Figure 2). Specifically, classes 7-10 each had training data less 20. Our models have learned to classify the counts much better than the baseline for classes with more data (i.e., classes 2-6). With more diverse training data, our methods are more likely to get better results on all the classes.

## 6    Conclusions

We carried out a reproducibility study of the paper 'Teaching CLIP to Count to Ten' (Paiss et al. 2023), that introduced a new loss function to fine-tune VLM's to make them adept at counting tasks and released a benchmark for the same. In our efforts to reproduce the paper, we modified the loss function for a small training dataset ($\sim 640$ times smaller than the dataset used in the paper). Despite the small amount of data our modifications have been shown to improve the performance of the model, surpassing the baseline. We also thoroughly audited the benchmark CountBench published in the paper, finding that $\sim 30$ of the 540

images in the dataset are missing. We have made our datasets and code public, in an effort to make the work on count-aware VLMs to be more accessible to the research community.

## References

Omer Bar-Tal, Hila Chefer, Omer Tov, Charles Herrmann, Roni Paiss, Shiran Zada, Ariel Ephrat, Junhwa Hur, Guanghui Liu, Amit Raj, Yuanzhen Li, Michael Rubinstein, Tomer Michaeli, Oliver Wang, Deqing Sun, Tali Dekel, and Inbar Mosseri. Lumiere: A space-time diffusion model for video generation, 2024.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable video diffusion: Scaling latent video diffusion models to large datasets, 2023.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. URL https://openai.com/research/video-generation-models-as-world-simulators.

Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolo, 1 2023. URL https://ultralytics.com.

Roni Paiss, Ariel Ephrat, Omer Tov, Shiran Zada, Inbar Mosseri, Michal Irani, and Tali Dekel. Teaching clip to count to ten, 2023.

Hieu Pham, Zihang Dai, Golnaz Ghiasi, Kenji Kawaguchi, Hanxiao Liu, Adams Wei Yu, Jiahui Yu, Yi-Ting Chen, Minh-Thang Luong, Yonghui Wu, Mingxing Tan, and Quoc V. Le. Combined scaling for zero-shot transfer learning, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021. URL https://arxiv.org/abs/2103.00020.

Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. LAION-400M: open dataset of clip-filtered 400 million image-text pairs. *CoRR*, abs/2111.02114, 2021. URL https://arxiv.org/abs/2111.02114.
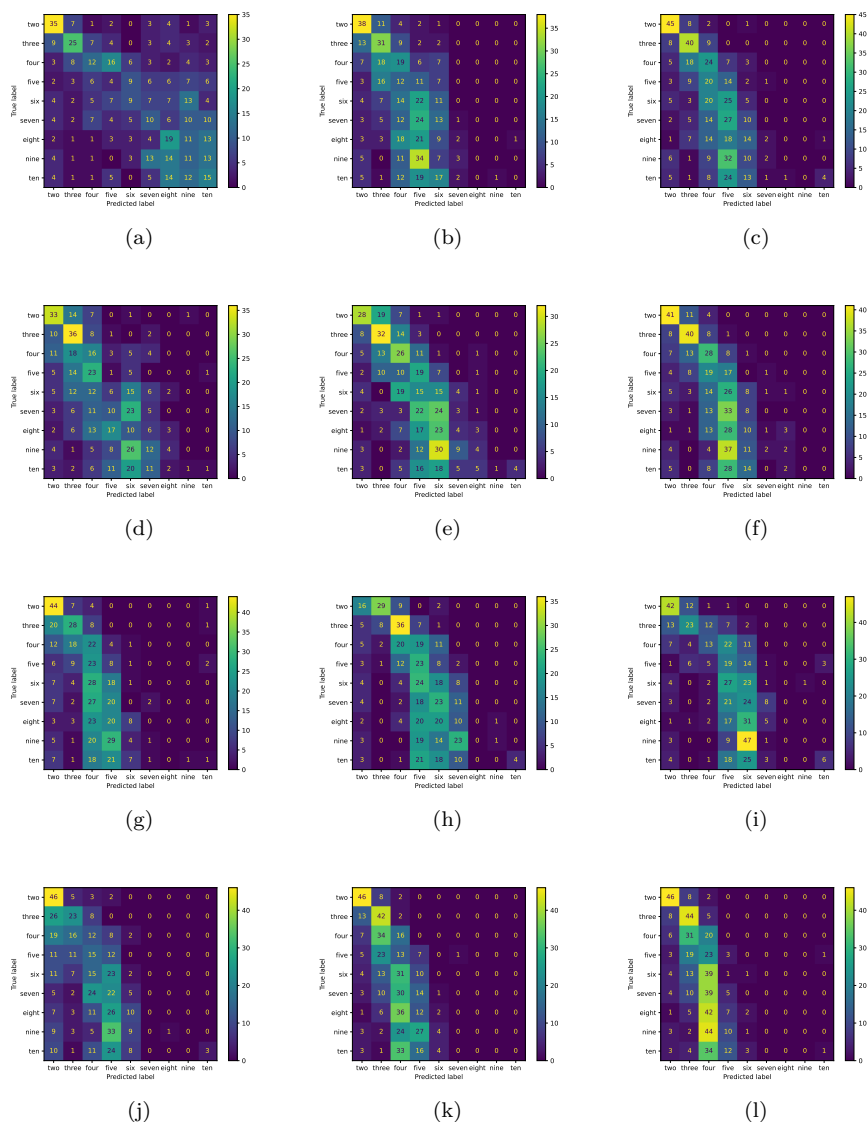
Figure 4: Confusion matrices for models trained till the end of the $10^{th}$ epoch for: (a)baseline (b) $\lambda = 1$ with no scheduler (c) scheduler and $\lambda = 1$ (base model) (d) $\lambda_{norm}$ (e) scheduler and $\lambda_{norm}$ (f) scheduler and $\lambda_{norm}$ and $L_{count+}$ (g) $\lambda_{modal}$ (h) scheduler and $\lambda_{modal}$ (i) scheduler and $\lambda_{modal}$ and $L_{count+}$ (j) $\lambda_{log}$ (k) scheduler and $\lambda_{log}$ (l) scheduler and $\lambda_{log}$ and $L_{count+}$

## A    Visualising Results

This section the contains the confusion matrices for all the experiments as mentioned in Table 1 in Section 4. Figure 4 contains the confusion matrices for models till the end of the 10th epoch. Figure 5 contains the confusion matrices for an early stopping mechanism selecting models with maximum validation accuracy.
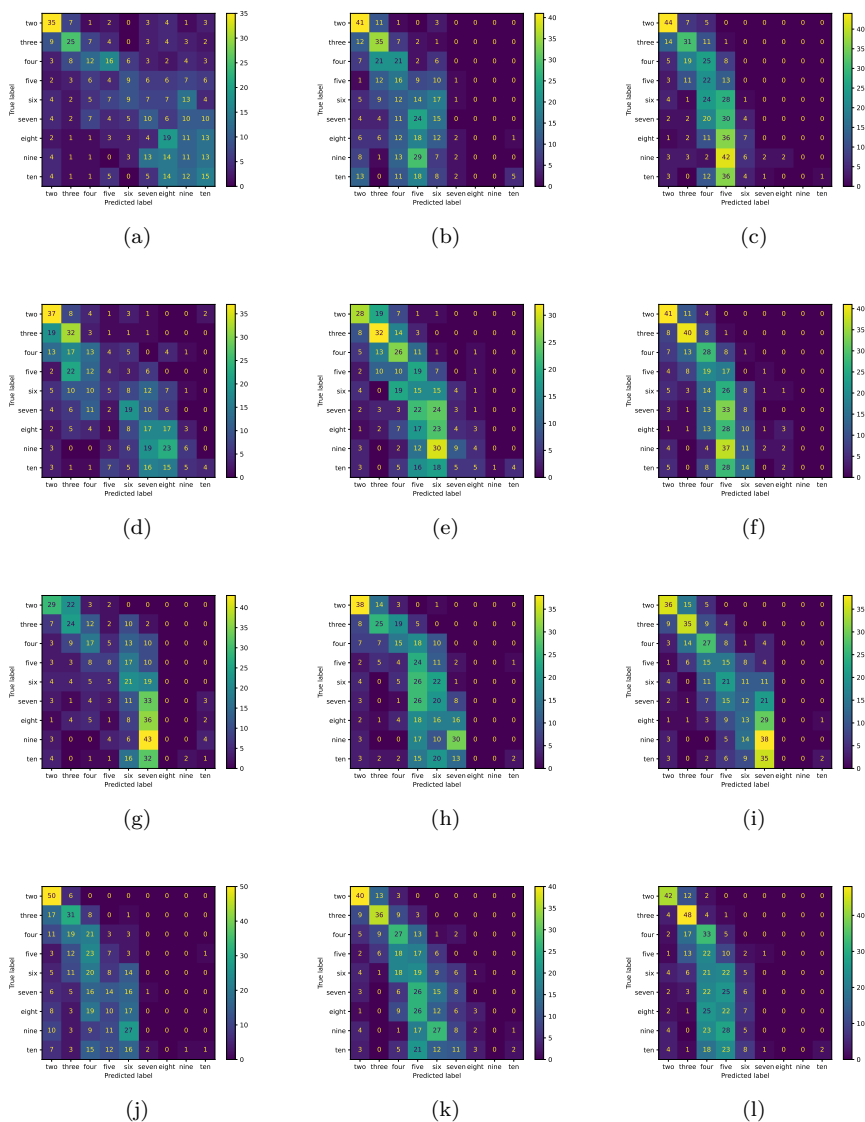
Figure 5: Confusion matrices for an early stopping mechanism selecting models with maximum validation accuracy for :(a)baseline (b) $\lambda = 1$ with no scheduler (c) scheduler and $\lambda = 1$ (base model) (d) $\lambda_{norm}$ (e) scheduler and $\lambda_{norm}$ (f) scheduler and $\lambda_{norm}$ and $L_{count+}$ (g) $\lambda_{modal}$ (h) scheduler and $\lambda_{modal}$ (i) scheduler and $\lambda_{modal}$ and $L_{count+}$ (j) $\lambda_{log}$ (k) scheduler and $\lambda_{log}$ (l) scheduler and $\lambda_{log}$ and $L_{count+}$