School of Computer Science
Faculty of Science and Engineering
University Of Nottingham
Malaysia

UG FINAL YEAR DISSERTATION REPORT

- Title -

Learning Generalisable Models for Quality Classification of Germinated Oil Palm Seeds

Student's name : John Chieng Xiang Hao

Student Number : 20300676

Supervisor Name : Dr Iman Yi Liao

Year : 2024

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF BACHELOR OF SCIENCE IN COMPUTER SCIENCE - ARTIFICIAL INTELLIGENCE BSc (HONS) THE UNIVERSITY OF NOTTINGHAM

- Title -

Learning Generalisable Models for Quality Classification of Germinated Oil Palm Seeds

Submitted in May 2024, in partial fulfillment of the conditions of the award of the degrees B.Sc

John Chieng Xiang Hao

School of Computer Science

Faculty of Science and Engineering

University of Nottingham

Malaysia

I hereby declare that this dissertation is all my own work, except as indicated in the text:

30/4/2024

Abstract

In Malaysia, the yearly requirement of oil palm planting materials is conservatively estimated to be 50 million.To dispatch high quality seeds, traditional oil palm seed production is laborious and prone to human error. Recently, the use of state-of-the-art Convolutional Neural Networks (CNNs) have been applied to the domain of plant phenotyping and have demonstrated superior performance in, e.g., classifying Crambe seed quality based on X-ray images and differentiating corn species. Some prelimenary results that have been published shown that a trained CNN model (both shallow and deep pretrained model) could acheive around 95-97% accuracy in both training and testing images of palm oil seed that were collected in the same batches, (i.e.,same environmental and camera conditions). However, when such a trained model is applied directly to images of germinated oil palm seeds collected in different batches, the model generalises poorly, e.g., the testing accuracy would drop by 15% to 30%. The issue is usually referred to as generalisability to external dataset and is common in almost all domains, e.g., medical image analysis. So in this project, We will explore the suitability of a vision language models specifically CLIP that leverages a contrastive learning approach that learns joint embeddings of images and text, allowing it to capture rich semantic representations in the task of classification of germinated palm oil seed.By leveraging prompt learning and prompt engineering techniques, we aim to tailor the input or prompts given to the CLIP model, enhancing its understanding and classification of germinated palm oil seeds.

1 Introduction

In Malaysia, the yearly requirement of oil palm planting materials is conservatively estimated to be 50 million [39].To dispatch high quality seeds, traditional palm oil seed production is laborious and prone to human error. Recently, the use of state-of-the-art Convolutional Neural Networks (CNNs) have been applied to the domain of plant phenotyping and have demonstrated superior performance in, e.g., classifying Crambe seed quality based on X-ray images [5] and differentiating corn species [18]. Some preliminary results of using CNNs in classifying germinated palm oil seed have been publish here [30] and here [29] have shown that that a trained CNN model (both shallow and deep pretrained models) could acheive around 95-97% accuracy in both training and testing on images that were collected in the same batch of germinted palm oil seed (i.e.,same environmental and camera conditions). However, when such a trained model is applied directly to images of germinated oil palm seeds collected in different batches, the model generalises poorly, e.g., the testing accuracy would drop by 15% to 30%. The issue is usually referred to as generalisability to external dataset and is common in almost all domains, e.g., medical image analysis [14].

Such issue could be caused by the common approach for building state-of-the-art visual recognition systems which is to train vision models to predict for a fixed set of object categories using discrete labels [16, 8]. From a technical

point of view, this is achieved by matching image features—produced by a vision model like ResNet[16] or ViT[8]—with a fixed set of weights that are seen as visual concepts and initialized randomly. Although training categories often have a textual form, such as "goodseed" or "badseed," they will be converted into discrete labels just for easing the computation of the cross-entropy loss, leaving the semantics encapsulated in texts largely unexploited. Such a learning paradigm limits visual recognition systems to closed-set visual concepts, making them unable to deal with new categories since additional data are required for learning a new classifier.

To overcome the limitations of traditional CNN models in generalizing to external datasets, we propose leveraging advanced vision-language models, specifically CLIP [37], in seed quality classification. Recently, vision-language pre-training such as CLIP [37] and ALIGN [20] has emerged as a promising alternative for visual representation learning. The main idea is to align images and raw texts using two separate encoders—one for each modality. For instance, both CLIP and ALIGN formulate the learning objective as a contrastive loss, which pulls together images and their textual descriptions while pushes away unmatched pairs in the feature space. By pre-training at a large scale, models can learn diverse visual concepts and can readily be transferred to any downstream task through prompting [37, 20, 11, 28, 41, 47]. In particular, for any new classification task one can first synthesize the classification weights by giving sentences describing task-relevant categories to the text encoder, and then compare with image features produced by the image encoder.

As such, in this project we will be leveraging the synergies of prompt engineering and prompt learning methodologies with CLIP as a vision language model to tackle the task of seed quality classification. By harnessing CLIP's inherent capacity to bridge the semantic gap between visual and textual modalities, we aim to overcome the limitaions of a traditional visual recognition system and create a robust and generalisable seed quality classification model.

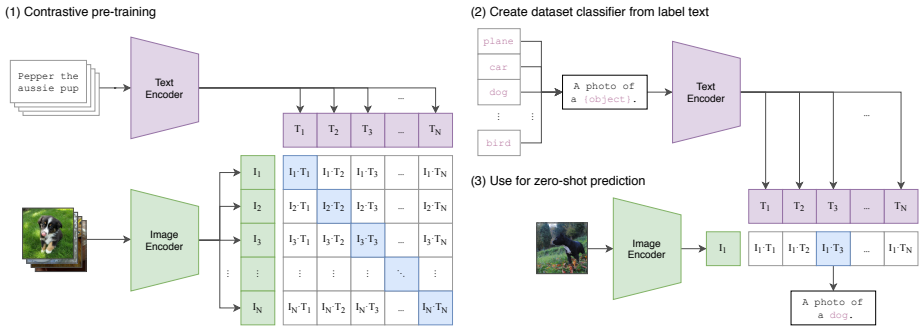


Figure 1: Summary of CLIP. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples.

2 Related Work

2.1 Vision-Language Models

Vision-language models have emerged as powerful tools for learning versatile visual representations and facilitating zero-shot transfer to a diverse array of downstream classification tasks through the mechanism of prompting [37, 20, 48, 41, 47]. Recent advancements in vision-language learning, notably exemplified by CLIP [37] and ALIGN [20], owe much of their progress to breakthroughs in three critical areas: i) text representation learning leveraging Transformer architecturess [43], ii) large-minibatch contrastive representation learning [3, 15, 17], and iii) access to extensive training datasets—CLIP draws from a pool of 400 million curated image-text pairs, while ALIGN capitalizes on 1.8 billion noisy image-text pairs.

While the concept of aligning images and text within a shared embedding space has been explored for nearly a decade [42, 10, 9], the methodologies have evolved significantly with advancements in technology. Early endeavors focused on text feature extraction predominantly relied on pre-trained word vectors [42, 10] or hand-crafted TF-IDF features [9, 23]. Matching image and text features has been approached through various techniques including metric learning [10], multi-label classification [22, 13], n-gram language learning [26], and more recently, captioning methodologies [6].

2.2 Prompt Learning

Prompt learning, pioneered by Petroni *et al.*[34], has garnered significant attention in Natural Language Processing (NLP) over recent years [31, 40, 24, 27, 21, 34]. This technique involves prefixing instructions to model inputs during pre-training, facilitating enhanced performance on downstream tasks. While manually defined prompts, as demonstrated by Petroni *et al.*[34] and Pörner *et al.* [35], have shown promise in improving language models, they may lack optimization and precision, potentially leading to suboptimal results. Consequently, various methods have emerged to automatically explore and identify optimal prompts [21, 40, 50], striving for more accurate model estimations. Notably, the integration of prompts into vision-language models has revolutionized the learning of generic visual representations [37, 19, 51]. Pioneering works such as ALIGN [19] and CLIP [37] stand out in this domain. CLIP, for instance, achieves state-of-the-art visual representation learning through pre-training on a vast dataset of 400 million image-text pairs. Additionally, Zhou *et al.*[51] introduce CoOp (Context Optimization), leveraging continuous representations for prompt modeling. CoOp offers a straightforward yet effective approach for adapting CLIP-like vision-language models for downstream image recognition tasks.

2.3 Prompt Engineering

Prompt engineering represents a pivotal approach for customizing large pre-trained models, often referred to as foundation models, to address novel tasks by enhancing the model input with task-specific cues. This augmentation is achieved through the incorporation of an additional component known as a prompt, which can take various forms such as manually crafted natural language instructions [45], automatically generated natural language instructions [49], or automatically generated vector representations [25]. These instructions are categorized as discrete prompts or *hard prompts* when expressed in natural language, while vector representations are termed continuous prompts or soft prompts.

The rise of prompt engineering has coincided with the emergence of large pre-trained models, marking a transformative shift in machine learning (ML) methodologies. Unlike the traditional paradigm, which necessitates extensive data labeling and either training task-specific ML models from scratch or fine-tuning pre-trained models, prompt engineering enables the adaptation of pre-trained models to specific tasks with minimal labeling effort. Moreover, it mitigates the need for extensive parameter tuning, facilitating the scalability and versatility of ML models across diverse tasks. By prompting pre-trained models with task-specific cues, prompt engineering leverages the existing knowledge encoded within these models to guide their performance on new tasks.

Depending on the nature of the provided hints, prompt engineering encompasses two primary approaches. If the hints are human-interpretable natural language instructions, the approach is often referred to as *In-Context Learning* [7], allowing models to learn from task instructions, demonstrations, or supporting context. Alternatively, when the hints are represented as continuous vector representations, the approach is termed *Prompt-Tuning* [25], wherein prompts are optimized directly within the embedding space of the model.

In comparison to the traditional paradigm, prompt engineering offers several advantages. Firstly, it necessitates only a small amount of labeled data to adapt a pre-trained model to new tasks, substantially reducing the need for human supervision and computational resources for fine-tuning. Secondly, prompt engineering empowers a pre-trained model to make predictions on new tasks solely based on the prompt, without necessitating updates to the model's parameters, thereby enabling the deployment of a single model across a wide range of downstream tasks. This capability renders large-scale pre-trained models applicable to real-world applications.

Prompt engineering initially gained traction in natural language processing (NLP) [32, 36] before garnering significant attention in computer vision [2, 44] and vision-language modeling [1, 46]. Despite the abundance of literature on prompt engineering in the NLP domain, a systematic overview to illuminate the current state of prompt engineering on pre-trained vision-language models (VLMs) is presently lacking, which present their own unique challenge.

3 Methodology

3.1 Data Preperation

For the dataset, We used the dataset provided by Advanced Agricultural Resources (AAR) Sdn. Bhd. The dataset consists of three batches of germinated palm oil seed taken in different environments. With the first batch which we used in training, second batch taken in a normal room lighting environment and lastly the third batch taken in a light box.

3.1.1 Image Augmentation

The original training dataset only consists of a total of 1752 training images, so in order to enhance the diversity of the original image dataset, we applied image augmentation techniques in order to introduce variability in the images while preserving their semantic content and labels. The following image augmentation techniques were employed:

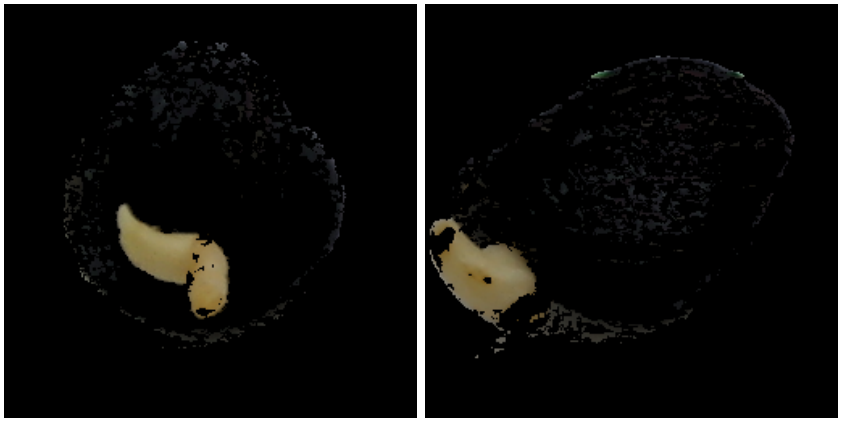
3.1.1.1 Color Segmentation

```
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
shoot_upper_boundary = np.array([175, 255, 170])
shoot_lower_boundary = np.array([20, 27, 23])
mask = cv2.inRange(hsv, shoot_lower_boundary, shoot_upper_boundary)
target = cv2.bitwise_and(image, image, mask=mask)
```

The code snippet performs the following operations:

- **Color Space Conversion**: Converts the input image from the BGR color space to the HSV (Hue, Saturation, Value) color space using the `cv2.cvtColor()` function. The HSV color space is better suited for color segmentation tasks as it separates color information from brightness.
- **Color Thresholding**: Defines upper and lower boundaries in the HSV color space to specify the range of colors to be segmented. The `shoot_upper_boundary` and `shoot_lower_boundary` arrays represent the upper and lower bounds of the beige color range of the germinated part of the palm oil seed, respectively.
- **Mask Creation**: Creates a binary mask using the `cv2.inRange()` function, which filters out pixels that fall within the specified color range defined by the upper and lower boundaries. The resulting mask highlights regions of interest that match the creamy beige color of the palm shoot.
- **Region Extraction**: Applies the binary mask to the original image using the `cv2.bitwise_and()` function to extract the regions of interest (i.e., the palm shoots) from the input image.

In order to highlight the germinated part of the seed color segmentation was performed.This code segments the germinated part of the seed, as from what we heard from the workers that is experienced with classifying good seed and bad seeds, one way of differentiating between a good and bad seed is the length and color of the plumule and radicle or shoot of the seed. Good Seed tend to have a more even shoot and have a creamy beige color to them while the bad seeds tend to have a more uneven and stunted shoot while having shoots with dark brown patches on them. By segmenting only the germinated part of the seed, hopefully we can direct the focus of the model more towards the germinated part of the seed rather than the background which does not contribute to the seed classification process.



(a) Segmented Good Seed Sample (b) Segmented Bad Seed Sample

Figure 2: Segmented Good and Bad Seed Samples

3.1.1.2 Random Transformations

```
transforms.RandomCrop(180),
transforms.ColorJitter(),
transforms.RandomHorizontalFlip(),
transforms.RandomVerticalFlip(),
transforms.RandomResizedCrop(180, scale=(0.8, 1.2), ratio=(1.0, 1.0)),
```

- **RandomCrop**: Randomly crops the input image to a size of 180x180 pixels, introducing variability in image composition.
- **ColorJitter**: Applies random color jittering transformations, including brightness, contrast, saturation, and hue adjustments, to simulate changes in lighting conditions and color appearance.

- **RandomHorizontalFlip**: Randomly flips the input image horizontally with a probability of 0.5, augmenting the dataset with horizontally mirrored versions of images.
- **RandomVerticalFlip**: Randomly flips the input image vertically with a probability of 0.5, augmenting the dataset with vertically mirrored versions of images.
- **RandomResizedCrop**: Randomly crops and resizes the input image to a size of 180x180 pixels, with random scaling and aspect ratio adjustments within specified ranges, simulating variations in image size and aspect ratio.

(a) Augmented Good Seed Sample (b) Augmented Bad Seed Sample

Figure 3: Augmented Good and Bad Seed Samples

3.1.1.3 Image Resolution Reduction

```
img = img.resize((96, 96), resample=Image.BILINEAR).convert('RGB')
w, h = img.size
q = random.randint(10, 70)
draw = ImageDraw.Draw(img)
draw.text(
    (random.randint(0, w//2), random.randint(0, h//2)),
    str(q),
    fill=(255, 255, 255)
)
```

The code snippet performs the following operations:

- **Image Resizing**: Resizes the input image to a size of 96x96 pixels using bilinear interpolation and converts it to the RGB color space to ensure consistency in dimensions and color channels.
- **Random Text Generation**: Generates a random integer value (‘q‘) between 10 and 70, which will be drawn on the image.
- **Drawing Text on Image**: Draws the randomly generated text (‘q‘) on the image at a random position within the top-left quarter of the image. The text color is set to white (RGB: 255, 255, 255) to ensure visibility against the background.

This transformation reduces the size of the image and causes a decrease in image quality. Ultimately, this aims to introduce variability into the dataset, which can increase the robustness of the dataset for the model.

(a) Quality Reduction Good Seed Sample (b) Quality Reduction Bad Seed Sample

Figure 4: Quality Reduction Good and Bad Seed Samples

3.1.1.4 Image Interpolation

```
original_height, original_width = img.shape[:2]

new_width = 500
new_height = 300

resized_image = cv2.resize(img, (new_width, new_height),
                           interpolation=cv2.INTER_CUBIC)
```

- **Original Image Dimensions Retrieval**: Retrieves the original height and width of the input image using the `shape` attribute of the image array.
- **New Dimensions Definition**: Defines the new dimensions for resizing the image. The `new_width` and `new_height` variables represent the desired width and height of the resized image, respectively.
- **Bicubic Interpolation Resizing**: Performs bicubic interpolation resizing of the input image using the `cv2.resize()` function with the `interpolation=cv2.INTER_CUBIC` parameter. Bicubic interpolation is a high-quality resizing method that preserves image details and reduces aliasing artifacts.
- **Conversion to PIL Format**: Converts the resized image back to the PIL (Python Imaging Library) format using the `Image.fromarray()` function. This conversion is necessary for compatibility with other image processing libraries and functions that require PIL format inputs.

This code snippet applies the bicubic interpolation on the images, it increases the images to match the image size from the batch 2 and batch 3 seeds. Hopefully, we can reduce the differences and promote consistency between batches and increase the generalisability of the model.

(a) Interpolated Good Seed Sample (b) Interpolated Bad Seed Sample

Figure 5: Interpolated Good and Bad Seed Samples

3.2 Model

CLIP comprises two distinct encoders: one for processing images and the other for handling text. The image encoder aims to transform high-dimensional images into a lower-dimensional embedding space, typically implemented using architectures like ResNet-50 [16] or Vision Transformer (ViT) [8]. Conversely, the text encoder utilizes a Transformer [43] architecture to generate representations from natural language. When processing text sequences, CLIP employs byte pair encoding (BPE) [38] to convert tokens, including punctuation, into unique numeric IDs, resulting in a vocabulary size of 49,152. Text sequences are augmented with `Start of Sequence (SOS)` and `End of Sequence (EOS)` tokens and are capped at a fixed length of 77 before being mapped to 512-dimensional word embedding vectors. These vectors are then fed into the Transformer, with subsequent layer normalization and linear projection layers applied to the features at the EOS token position.

CLIP [37] is trained with image-text pairs in a contrastive manner. Each input text describes a category in the format of "a photo of a [CLASS]" ([CLASS] is the class token). A positive pair is an image x_i with its corresponding text t_i describing the category of x_i. A negative pair is an image x_i with an irrelevant description $t_j, j \neq i$ in the mini-batch. The training objective is to maximize the cosine similarity of positive pairs and minimize the cosine similarity of negative pairs. The contrastive learning objective aligns the image and text representation in the same feature space.

Since CLIP is pre-trained to predict whether an image matches a textual description, it naturally fits zero-shot recognition. This is achieved by comparing image features with the classification weights synthesized by the text encoder, which takes as input textual descriptions specifying classes of interest. Formally, let $\boldsymbol{f}$ be image features extracted by the image encoder for an image $\boldsymbol{x}$ and $\{\boldsymbol{w}_i\}_{i=1}^K$ a set of weight vectors generated by the text encoder. K denotes the number of classes and each $\boldsymbol{w}_i$ is derived from a prompt that could have the form of "a photo of a [CLASS]." where the class token is replaced by the specific class name, such as "cat," "dog" or "car." The prediction probability is then computed as

$$p(y = i|\boldsymbol{x}) = \frac{\exp(\cos(\boldsymbol{w_i}, \boldsymbol{f})/\tau)}{\sum_{j=1}^{K} \exp(\cos(\boldsymbol{w_j}, \boldsymbol{f})/\tau)}$$

where τ is a temperature parameter learned by CLIP and $\cos(\cdot, \cdot)$ denotes cosine similarity.

Compared with the traditional classifier learning approach where closed-set visual concepts are learned from random vectors, vision-language pre-training allows open-set visual concepts to be explored through a high-capacity text encoder, leading to a broader semantic space and in turn making the learned representations more transferable to downstream tasks.

For the model used in the experiment, as we have stated above a model from CLIP would be used. However, instead of directly using CLIPs implementation of the model, we will be using open_clip [4] an open source implementation

of CLIP which out performs the original implementation of the open sourced models for CLIP. The model we will be fine tuning will be the **Vit-B-32** model with the pretrained weight of [datacomp_xl_s13b_b90k].

3.3 Training

Although CLIP demonstrates impressive zero-shot transfer learning capabilities, fine-tuning is necessary for our task of classifying germinated palm oil seeds. This requirement arises because CLIP is trained on 400 million curated image-text pairs from the internet and may lack familiarity with specialized domains such as palm oil seed germination. Consequently, to adapt the model to our dataset and improve its performance on our task, fine-tuning becomes necessary.

The loss function for our fine tuning process is cross-entropy loss between CLIP's predictions and the ground truth labels for both image and text inputs.

The hyperparameters used for all the training are fixed, as we train the model for 30 epochs with a batch size of 64 using the ADAM optimizer with β_1 of 0.9 ,β_2 of 0.98 and ϵ of $1e-6$ as its parameter. The initial learning rate is set to $5e-5$ with a cosine annealing learning rate scheduler.

For the prompts, we apply first apply a manual prompt engineering method by manually crafting the prompts and fine tuning the model until we get the best accuracy on the datasets before combining it with prompts generated from prompt learning.

The prompt learning method we used for this is known as the Context Optimization or (CoOp) by Zhou *et al.*[51] which avoids manual prompt tuning by modeling context words with continuous vectors that are end-to-end learned from data while the massive pre-trained parameters are frozen. An overview is shown in Figure 6.

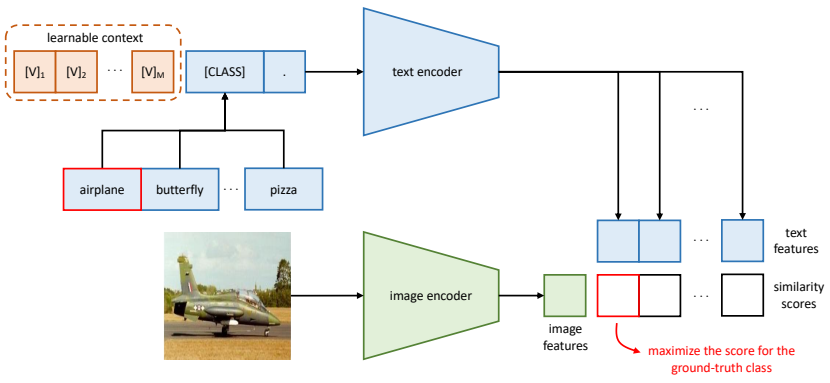


Figure 6: **Overview of Context Optimization (CoOp)**. The main idea of CoOp is to model a prompt's context using a set of learnable vectors, which can be optimized through minimizing the classification loss.

In the end our prompt would be:

$$\overbrace{\underbrace{\text{A photo of a germinated palm oil}}_{\text{Prefix}} + \underbrace{\text{Context}}_{\text{Prompt Engineering}} + \underbrace{\text{Context}}_{\text{Prompt Learning}} + \underbrace{\text{Class}}_{\text{Class}}}^{\textbf{Prompt}}$$

4 Experiment

4.1 Zero-Shot Transfer

As discussed above, CLIP boasts an excellent zero shot transfer learning ability, and that is what we explored first. However the results are rather underwhelming, as CLIP although known for its ability to do zero shot classification on never seen before datasets it fails to perform when it comes to more specialized domain, which in our case is germinated palm oil seed classification.

As we can see from Table. 1 and 2, when Good Seed is set as the first class in the classification, we get a skewed result where all the seeds are classified as good seed and none are being classified as bad seed, vice versa. The prompt we used are is crafted with the template given by CLIP's creator which stated that with the use of the template of "`A photo of a {label}.`" they saw a accruacy increase of 1.3% on ImageNet rather than using just the template. So the prompt we used for this classification task is:

$$\overbrace{\text{A photo of a good germinated palm oil seed}}^{\textbf{Good Seed Prompt}} \tag{1}$$

$$\overbrace{\text{A photo of a bad germinated palm oil seed}}^{\textbf{Bad Seed Prompt}} \tag{2}$$

Class	Recall	Precision	F1 Score
Good Seed	1.0	0.51	0.67
Bad Seed	0.0	1.00	0.00

Table 1: Zero-Shot Result with Good Seed as First Class

Class	Recall	Precision	F1 Score
Good Seed	0.005	0.50	0.67
Bad Seed	0.98	0.30	0.01

Table 2: Zero-Shot Result with Bad Seed as First Class

4.2 Prompt Engineering

In our prompt engineering process, we initially crafted prompts manually for each class and iteratively refined them based on testing results. It's important to note that we didn't include all iterations of our prompts in this report due to space constraints. Instead, we focus on iterations where we observed interesting or improved results. The first-generation prompts we used were the same as those utilized for zero-shot transfer learning discussed earlier 1, 2. Results for all three batches of seeds are provided below. It's worth mentioning that the interpolation transformation was not applied to the dataset during these tests; this transformation was introduced later in the experiment. Nonetheless, these results offer valuable insights into our prompt refinement process and decision-making.

Transformation	Accuracy	Pretrained Weight
None	0.91	No
1	0.89	No
1,2	0.91	No
1,2,3	0.92	No
1,2,3	0.98	datacomp_xl_s13b_b90k

Table 3: **Result for Test Set**. Transformation Notations: 1 - Transformations 3.1.1.1, 2 - Color Segmentation 3.1.1.2, 3 - Image Resolution Reduction 3.1.1.3.

Transformation	Accuracy	Pretrained Weight
None	0.76	No
1	0.77	No
1,2	0.80	No
1,2,3	0.81	No
1,2,3	0.87	datacomp_xl_s13b_b90k

Table 4: **Result for Normal Room Lighting Set**. Transformation Notations: 1 - Color Segmentation 3.1.1.1, 2 - Transformations 3.1.1.2, 3 - Image Resolution Reduction 3.1.1.3.

Transformation	Accuracy	Pretrained Weight
None	0.58	No
1	0.60	No
1,2	0.644	No
1,2,3	0.66	No
1,2,3	0.77	datacomp_xl_s13b_b90k

Table 5: **Result for Light Box**. Transformation Notations: 1 - Color Segmentation 3.1.1.1, 2 - Transformations 3.1.1.2, 3 - Image Resolution Reduction 3.1.1.3.

As evident from the results, the model exhibits excellent performance on the test dataset (batch 1), but its performance declines significantly when faced with unseen data such as batches 2 and 3, with a drop ranging from 11% to 21%. This drop in performance can be attributed to the lack of contextual information during model training, as highlighted by [33]. Despite advancements in classification performance, large models like CLIP still struggle to identify cues that humans can easily recognize. For instance, when identifying a hen, humans typically look for visual cues such as its beak, feathers, and other distinguishing features, which may not be inherently understood by machine learning models. To address this issue, we propose incorporating visual cues or context into the prompts, such as color, shape, and size characteristics of the seeds. By including these cues, we aim to encourage the model to consider these visual attributes when classifying items. Drawing from the experience of workers involved in seed quality classification, we recognize the significance of the growth and coloration of the plumule and radicle in the classification task. Therefore, we refined our prompt to incorporate these crucial visual cues:

A photo of a good germinated palm oil seed

+

$$\underbrace{\text{that exhibits uniform growth in both its plumule and radicle without any signs of discoloration}}_{\textbf{Good Seed Context}}$$

A photo of a bad germinated palm oil seed

+

$$\underbrace{\text{with irregular growth in both its plumule and radicle with signs of discoloration}}_{\textbf{Bad Seed Context}}$$

With the inclusion of additional context to distinguish between good and bad seeds, although there is no noticeable performance enhancement in the first two batches, a 2% increase in accuracy is evident in the third batch. This suggests an improvement in the model's robustness or generalizability. It is crucial to note that all these findings are based on testing conducted without the interpolation transformation applied yet, this is because the interpolation transformation was only applied at the end of the experiment:

Batch	Accuracy
Test	0.97
Normal Room Lighting	0.87
Light Box	0.79

Table 6: Results on all three batches

Upon conducting additional testing, we found that simplifying the terminology and providing more detailed descriptions or context significantly enhances the model's performance. For instance, replacing "plumule" and "radicle" with the term "shoot" to encompass both, and delving into finer distinctions such as physical color, notably improves results. Specifically, we describe the shoots of good seeds as having a creamy beige hue, while those of bad seeds exhibit dark brown patches. This refinement of language and specificity contributes to a notable improvement in the model's accuracy.

A photo of a germinated palm oil seed

+

$$\underbrace{\text{with shoot of uneven length, distinguished by the presence of brown patches on the shoot, indicating a good seed}}_{\textbf{Good Seed Improved Context}}$$

or

$$\underbrace{\text{with shoot of even length, distinguished by the display of a creamy beige color on the shoot, indicating a good seed}}_{\textbf{Bad Seed Improved Context}}$$

As you can observe in this prompt, the class is placed at the end of our prompts, that is to ensure the fluidity of the prompt.With the enhanced and more elaborate context provided, we witness notable improvements, amounting to approximately 3% for both the Normal Room Lighting and Light Box batches. 7.

Batch	Accuracy
Test	0.96
Normal Room Lighting	0.90
Light Box	0.81

Table 7: Results on all three batches

4.2.1 With Interpolation Applied

As previously mentioned, all experiments conducted thus far were executed without the application of the interpolation transformation. However, upon implementing the interpolation transformation, we encountered unexpected outcomes. Surprisingly, utilizing just the base prompt led to superior performance, surpassing all previous prompts mentioned earlier. The results are presented below for reference 8.

Batch	Accuracy
Test	0.95
Normal Room Lighting	0.897
Light Box	0.82

Table 8: Results on all three batches with the base prompts

We see an overall increase for about 2% to 3%, which is a significant improvement in performance considering the fact that it is just the base prompt

without any context tokens at all. It appears that the interpolation transformation played a crucial role, as scaling the images from the training dataset to match those of the normal room lighting and light box environments allowed the model to generalize more effectively. However, when employing the optimal prompt identified previously, we no longer observe performance gains. In fact, there is even a decline in performance observed in the Normal Room Lighting dataset, indicating a potential issue with overfitting. 9. Regrettably, our testing of the prompts reached its conclusion due to time constraints. Nevertheless, we consider the results obtained thus far to serve as a solid baseline for future exploration. With additional time dedicated to crafting more refined prompts, we are confident that further enhancements can be made to optimize the performance of this model.

Batch	Accuracy
Test	0.96
Normal Room Lighting	0.8877
Light Box	0.82

Table 9: Results on all three batches with our best prompt

4.3 Prompt Learning

For the prompt learning method, we used the code provided by Zhou *et al.*[51] specifically the unified context version of prompt learning. The structure of the prompts are as follows :

$$[\mathrm{V}]_1[\mathrm{V}]_2 \dots [\mathrm{V}]_M[\mathrm{CLASS}]$$

where each $[\mathrm{V}]_m$ $(m \in \{1, \dots, M\})$ is a vector with the same dimension as word embeddings (i.e., 512 for CLIP), and M is a hyperparameter specifying the number of context tokens.Instead of the class being in the end, CoOp also offer a prompt structure with class at the end like so:

$$[\mathrm{V}]_1 \dots [\mathrm{V}]_{\frac{M}{2}}[\mathrm{CLASS}][\mathrm{V}]_{\frac{M}{2}+1} \dots [\mathrm{V}]_M$$

For the hyperparameters, we utilized Vit-B-32 as our backbone architecture, trained for 50 epochs with a learning rate of 0.002 and the SGD optimizer, coupled with a cosine annealing learning rate scheduler. We set the context length to 16 and the number of shots in the few-shot learning to 16. During experimentation, we explored both placing the class token at the end and at the middle of the prompts. For this particular experiment, rather than training on the training set and applying it on the Normal Room Lighting and Light Box datasets, we directly trained the model on these datasets. This approach was adopted to extract useful context tokens that could potentially be incorporated into our own prompts.

Class Token Placement	Accuracy
Middle	0.91
End	0.90

Table 10: Accuracy on the Test Dataset

Class Token Placement	Accuracy
Middle	0.88
End	0.89

Table 11: Accuracy on the Normal Room Lighting Dataset

Class Token Placement	Accuracy
Middle	0.78
End	0.75

Table 12: Accuracy on the Light Box Dataset

With one of the functions in CoOp, we are able to see the top 3 nearest words for each context token.

4.3.1 Test Set

Context No.	**Top Three Tokens**	**Nearest Value(Smaller is Better)**
1	['trucks', 'inking', 'nassau']	['0.8329', '0.8332', '0.8336']
2	['½', 'jana', 'cht']	['0.7316', '0.7379', '0.7388']
3	['forefront', 'sai', 'findyour']	['0.8401', '0.8447', '0.8448']
4	['¹', 'alizing', 'level']	['0.7528', '0.7530', '0.7534']
5	['chik', 'victims', 'continue']	['1.1209', '1.1220', '1.1222']
6	['gl', 'gag', 'anza']	['1.1083', '1.1160', '1.1171']
7	['lighter', 'tta', 'petit']	['1.1132', '1.1166', '1.1176']
8	['publishes', 'secured', 'brink']	['0.8618', '0.8658', '0.8658']
9	['gentlemen', 'stin', 'children']	['1.1531', '1.1547', '1.1550']
10	['stas', 'prepares', 'cas']	['0.8257', '0.8262', '0.8262']
11	['tallade', 'wak', 'apor']	['0.8000', '0.8010', '0.8032']
12	['pose', 'ji', 'selfi']	['0.6215', '0.6235', '0.6292']
13	['lywood', 'holla', 'itchen']	['1.0855', '1.0876', '1.0901']
14	['jas', 'champ', 'boo']	['0.8404', '0.8449', '0.8452']
15	['gie', 'vin', 'antique']	['0.6578', '0.6587', '0.6602']
16	['aaaaa', 'aaa', 'aaaaaa']	['0.6721', '0.6741', '0.6776']

Table 13: Context Tokens for Test Set, Class at the Middle Structure

Context No.	Top Three Tokens	Nearest Value(Smaller is Better)
1	['brilliant', 'tx', 'abound']	['0.5267', '0.5312', '0.5322']
2	['lane', 'aire', '6']	['0.5040', '0.5041', '0.5052']
3	['ðŁij¹', 'ss', 'staying']	['0.5449', '0.5473', '0.5478']
4	['hrs', 'months', 'hours']	['0.5215', '0.5221', '0.5227']
5	['championships', 'actress', 'skates']	['0.6017', '0.6027', '0.6031']
6	['ĩ', 'ator', 're']	['0.6295', '0.6318', '0.6319']
7	['chis', 'atown', 'jelly']	['0.5679', '0.5703', '0.5710']
8	['bub', 'karl', 'karl']	['0.7718', '0.7741', '0.7771']
9	['Ļ', 'elk', '$$']	['0.5193', '0.5288', '0.5291']
10	['o', 'ile', 'wanted']	['0.6165', '0.6183', '0.6190']
11	['wak', 'borne', 'unex']	['0.6002', '0.6117', '0.6132']
12	['harbour', 'importantly', 'instructions']	['0.5658', '0.5674', '0.5723']
13	['gotta', 'following', 'owner']	['0.5987', '0.6092', '0.6127']
14	['yall', 'dotted', 'tol']	['0.5287', '0.5344', '0.5355']
15	['âĊĭ¸ıàĊĭ¸ıâĊĭ¸ı', 'ðŁĴ', 'âĻ¥']	['0.5167', '0.5171', '0.5189']
16	['dow', 'awhile', 'park']	['0.6074', '0.6075', '0.6088']

Table 14: Context Tokens for Test Set, Class at the End Structure

4.3.2 Normal Room Lighting

Context No.	Top Three Tokens	Nearest Value(Smaller is Better)
1	['2', 'legend', 'kil']	['0.5796', '0.5850', '0.5861']
2	['celebrate', 'diwali', '6']	['0.6270', '0.6294', '0.6308']
3	['5', 'scents', '6']	['0.6027', '0.6049', '0.6062']
4	['shored', 'takeme', 'cancels']	['0.8981', '0.9053', '0.9061']
5	['great', 'apro', 'came']	['0.5998', '0.6017', '0.6024']
6	['rav', 'ı', 'kin']	['0.6661', '0.6702', '0.6772']
7	['xb', 'ents', 'ranbir']	['0.8383', '0.8393', '0.8406']
8	['patience', 'neat', 'boxes']	['0.7000', '0.7016', '0.7046']
9	['ra', 'osu', 'para']	['0.8602', '0.8625', '0.8630']
10	['ðŁĩ³ðŁĩ', 'digital', 'ðŁıIJ']	['0.7070', '0.7168', '0.7198']
11	['air', 'light', 'forme']	['0.5534', '0.5547', '0.5566']
12	['ji', 'hello', 'deter']	['0.5772', '0.5828', '0.5882']
13	['jewellery', 'houses', 'owner']	['0.4857', '0.4968', '0.4995']
14	['yall', 'tol', 'nireland']	['0.5123', '0.5176', '0.5206']
15	['zz', 'zi', 'stocks']	['0.5636', '0.5651', '0.5653']
16	['ville', 'hill', 'yol']	['0.5163', '0.5216', '0.5236']

Table 15: Context Tokens for Normal Room Lighting Set, Class at the End Structure

Context No.	Top Three Tokens	Nearest Value(Smaller is Better)
1	['2', 'legend', 'kil']	['0.5796', '0.5850', '0.5861']
2	['celebrate', 'diwali', '6']	['0.6270', '0.6294', '0.6308']
3	['5', 'scents', '6']	['0.6027', '0.6049', '0.6062']
4	['shored', 'takeme', 'cancels']	['0.8981', '0.9053', '0.9061']
5	['great', 'apro', 'came']	['0.5998', '0.6017', '0.6024']
6	['rav', 'ı', 'kin']	['0.6661', '0.6702', '0.6772']
7	['xb', 'ents', 'ranbir']	['0.8383', '0.8393', '0.8406']
8	['patience', 'neat', 'boxes']	['0.7000', '0.7016', '0.7046']
9	['ra', 'osu', 'para']	['0.8602', '0.8625', '0.8630']
10	['ðŁĩ³ðŁĩ', 'digital', 'ðŁıIJ']	['0.7070', '0.7168', '0.7198']
11	['air', 'light', 'forme']	['0.5534', '0.5547', '0.5566']
12	['ji', 'hello', 'deter']	['0.5772', '0.5828', '0.5882']
13	['jewellery', 'houses', 'owner']	['0.4857', '0.4968', '0.4995']
14	['yall', 'tol', 'nireland']	['0.5123', '0.5176', '0.5206']
15	['zz', 'zi', 'stocks']	['0.5636', '0.5651', '0.5653']
16	['ville', 'hill', 'yol']	['0.5163', '0.5216', '0.5236']

Table 16: Context Tokens for Test Set, Class at the End Structure

4.3.3 Light Box

Context No.	Top Three Tokens	Nearest Value(Smaller is Better)
1	['feast', 'brilliant', '2']	['0.5282', '0.5284', '0.5291']
2	['ine', 'bes', 'ðŁĴ']	['0.5116', '0.5161', '0.5165']
3	['psi', 'lists', 'creating']	['0.5403', '0.5416', '0.5419']
4	['months', **'wilt'**, 'illin']	['0.5479', '0.5481', '0.5482']
5	['hahahaha', 'tournament', 'ðŁĺĶ']	['0.6309', '0.6341', '0.6346']
6	['wickets', 're', 'ı']	['0.6026', '0.6047', '0.6067']
7	['chis', 'agencies', 'ppl']	['0.6750', '0.6760', '0.6761']
8	['boxes', 'b', '2']	['0.5276', '0.5299', '0.5305']
9	['sal', 'du', 'so']	['0.5724', '0.5801', '0.5812']
10	['nothing', 'container', 'juices']	['1.2572', '1.2582', '1.2618']
11	['alerts', 'wonder', 'forme']	['0.5309', '0.5318', '0.5344']
12	['ji', 'hello', 'lovely']	['0.5331', '0.5332', '0.5371']
13	['jewellery', 'boxes', 'gotta']	['0.4614', '0.4770', '0.4785']
14	['yall', 'tol', 'sway']	['0.5313', '0.5401', '0.5420']
15	['foryou', 'zi', 'lan']	['0.5086', '0.5105', '0.5110']
16	['zie', 'ville', 'divers']	['0.5322', '0.5343', '0.5396']

Table 17: Context Tokens for Light Box, Class in the Middle Structure

Context No.	Top Three Tokens	Nearest Value(Smaller is Better)
1	['brilliant', '2', 'tx']	['0.5163', '0.5177', '0.5186']
2	['bes', 'ine', 'lane']	['0.4918', '0.4943', '0.4944']
3	['6', '5', 'swc']	['0.5049', '0.5091', '0.5094']
4	['ori', 'hrs', 'urs']	['0.5239', '0.5269', '0.5289']
5	['stripes', 'prepares', 'go']	['0.5660', '0.5679', '0.5698']
6	['wickets', 'ward', 'wa']	['0.5492', '0.5497', '0.5521']
7	['chis', 'offerings', 'smiths']	['0.5229', '0.5332', '0.5350']
8	['b', 'Į', 'boxes']	['0.5230', '0.5268', '0.5305']
9	['Ļ', 'ðŁijī', 'Ĺ']	['0.5187', '0.5297', '0.5301']
10	['ðŁĩ³ðŁĩ', 'ile', 'i']	['0.5472', '0.5537', '0.5545']
11	['light', 'number', 'ad']	['0.5285', '0.5302', '0.5305']
12	['ji', 'harbour', 'hello']	['0.5054', '0.5144', '0.5145']
13	['jewellery', 'boxes', 'facing']	['0.4809', '0.4912', '0.4926']
14	['yall', 'tol', 'tomorrow']	['0.5026', '0.5130', '0.5156']
15	['zi', 'zz', 'funny']	['0.4856', '0.4903', '0.4945']
16	['ville', 'yol', 'yah']	['0.6138', '0.6200', '0.6213']

Table 18: Context Tokens for Light Box, Class at the End Structure

As you can see from the context tokens above, while prompt learning presents a valuable approach, it often yields extraneous or nonsensical prompts, failing to provide meaningful insights into the model's classification process. Nonetheless, we conducted experiments with certain context tokens that we felt was related to our classification task, such as 'wilt,' extracted from the Light Box dataset to describe our classification of bad seeds 17. By incorporating the 'wilt' token into our prompt, we devised a prompt that combines elements of both prompt learning and prompt engineering:

A photo of a germinated palm oil seed

+

$$\underbrace{\text{with shoot of even length, distinguished by the display of a creamy beige color on the shoot, indicating a good seed}}_{\textbf{Prompt Engineering}}$$

or

$$\underbrace{\text{with shoot of uneven length, distinguished by the presence of dark brown patches on the shoot}}_{\textbf{Prompt Engineering}}$$

$$\underbrace{\text{and showing signs of wilting}}_{\textbf{Prompt Learning}}\underbrace{\text{, indicating a bad seed}}_{\textbf{Prompt Engineering}}$$

However, the model performs poorly with this:

Batch	Accuracy
Test	0.94
Normal Room Lighting	0.83
Light Box	0.79

Table 19: Prompt Engineering + Prompt Learning Performance

Due to time constrains, we were unable to further test other combinations of the prompts. However despite encountering time constraints, our exploration of the interplay between prompt engineering and prompt learning remains promising. While our testing scope may have been limited, we recognize the vast potential inherent in synergizing human insights with machine-driven observations. This collaborative approach holds the key to crafting prompts that intricately capture the nuances and subtleties essential for the model's precise differentiation between good and bad seeds.

4.3.4 Text Augmentation

In an effort to enhance performance and mitigate overfitting, we employed text augmentation techniques with our top-performing prompts 7. Specifically, we utilized backtranslation to generate diverse captions for the images. Leveraging DeepL Translation, we translated the prompts into multiple languages including French, Spanish, Italian, and Portuguese, and subsequently back into English. This process aimed to expand the range of captions associated with each image, thus increasing dataset variability and linguistic diversity. However, the results yielded were underwhelming, leading us to discontinue this approach. Nonetheless, it represents a potential avenue for further exploration and refinement.

4.3.5 Domain Adaptation

We also explored domain adaptation as a strategy to enhance the model's performance. In this approach, the model was initially trained on the train dataset and then adapted from the source domain of Normal Room Lighting to the target domain of Light Box. While we observed an overall improvement in performance, we ultimately deemed this approach suboptimal.

Our objective is to develop a model that excels when trained solely on the base dataset, rather than one that relies on adaptation to unseen datasets for improved performance. As such, while domain adaptation may offer short-term performance gains, it does not align with our long-term goal of creating a robust and self-sufficient model capable of accurate classification across varying conditions.

Moving forward, our focus remains on refining the base model to achieve superior performance without the need for domain adaptation. By prioritizing the

development of a strong foundation, we aim to ensure the model's effectiveness and reliability across diverse environments and datasets.

5 Conclusion

In conclusion, our experimentation has demonstrated the significant potential of CLIP in the image classification task, particularly in the domain of seed quality classification. We have shown that CLIP can serve as the foundation for building robust and generalizable models capable of accurately categorizing seed quality.

Regrettably, our exploration has reached its current limit within this experiment. Nevertheless, we view our findings as a promising starting point for future enhancements and refinements. Further research and development efforts can capitalize on our groundwork to refine methodologies, explore additional applications, and advance the capabilities of CLIP-based models in diverse domains.

Overall, our results underscore the promise of CLIP as a versatile and effective tool in image classification tasks, with ample opportunities for continued innovation and improvement.

6 Future Work

- **Prompt Engineering**.Prompt engineering has proven instrumental in achieving satisfactory performance, although at the end, the base prompts were out performing our top-performing prompts due to the introduction of the interpolation transformation and likely causing overfitting, however we have shown that by refining the prompts, we were able to achieve a better performance.Hence, this underscores the need for further refinement in constructing prompts that can enhance performance without exacerbating overfitting.

- **Prompt Learning.** While our experiment did not extensively explore prompt learning, we recognize its considerable potential, as highlighted by Zhou *et al.*[51]. One avenue for further investigation could involve exploring class-specific context rather than a unified context. Additionally, Zhou *et al.* [52] work on Conditional Context Optimization (CoCoOp), presents another intriguing possibility for future exploration.

- **Domain Adaptation**. While we initially expressed reservations about the effectiveness of the domain adaptation approach, we recognize that its potential merits warrant further consideration. A notable project, 'Domain Adaptation via Prompt Learning' [12], exemplifies how CLIP can be leveraged for domain adaptation through prompt learning. This project offers valuable insights into the feasibility and effectiveness of domain adaptation strategies within the context of CLIP-based models

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

[2] Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. Exploring visual prompts for adapting large-scale models. *arXiv preprint arXiv:2203.17274*, 1(3):4, 2022.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

[4] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. *arXiv preprint arXiv:2212.07143*, 2022.

[5] A. D. de Medeiros, R. C. Bernardes, L. J. da Silva, B. A. L. de Freitas, D. C. F. dos Santos Dias, and C. B. da Silva. Deep learning based approach using x-ray images for classifying crambe abyssinica seed quality. *Industrial Crops and Products*, 164:113 – 378, 2021.

[6] Karan Desai and Justin Johnson. Virtex: Learning visual representations from textual annotations. In *CVPR*, 2021.

[7] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[9] Mohamed Elhoseiny, Babak Saleh, and Ahmed Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In *ICCV*, 2013.

[10] Andrea Frome, Greg Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In *NeurIPS*, 2013.

[11] Andreas Fürst, Elisabeth Rumetshofer, Viet Tran, Hubert Ramsauer, Fei Tang, Johannes Lehner, David Kreil, Michael Kopp, Günter Klambauer, Angela Bitto-Nemling, et al. Cloob: Modern hopfield networks with infoloob outperform clip. *arXiv preprint arXiv:2110.11316*, 2021.

[12] Chunjiang Ge, Rui Huang, Mixue Xie, Zihang Lai, Shiji Song, Shuang Li, and Gao Huang. Domain adaptation via prompt learning. *arXiv: 2202-06687*, 2022.

[13] Lluis Gomez, Yash Patel, Marçal Rusiñol, Dimosthenis Karatzas, and CV Jawahar. Self-supervised learning of visual features through embedding images into text topic spaces. In *CVPR*, 2017.

[14] A.-R. Habib, Y. Xu, K. Bock, and et al. Evaluating the generalizability of deep learning image classification algorithms to detect middle ear disease using otoscopy. *Scientific reports*, 13(1):5368, 2023.

[15] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[17] Olivier J. Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi, Carl Doersch, S. M. Ali Eslami, and Aäron van den Oord. Data-efficient image recognition with contrastive predictive coding. In *ICML*, 2020.

[18] S. Javanmardi, S.-H. Miraei Ashtiani, F. J. Verbeek, and A. Martynenko. Computer-vision classification of corn seed varieties using deep convolutional neural network. *Journal of Stored Products Research*, 92:101 – 800, 2021.

[19] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, volume 139, pages 4904–4916, 2021.

[20] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, 2021.

[21] Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. How can we know what language models know. *TACL*, 8:423–438, 2020.

[22] Armand Joulin, Laurens Van Der Maaten, Allan Jabri, and Nicolas Vasilache. Learning visual features from large weakly supervised data. In *ECCV*, 2016.

[23] Jimmy Lei Ba, Kevin Swersky, Sanja Fidler, et al. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *ICCV*, 2015.

[24] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv: 2104.08691*, 2021.

[25] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[26] Ang Li, Allan Jabri, Armand Joulin, and Laurens van der Maaten. Learning visual n-grams from web data. In *ICCV*, 2017.

[27] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL/IJCNLP*, pages 4582–4597, 2021.

[28] Yangguang Li, Feng Liang, Lichen Zhao, Yufeng Cui, Wanli Ouyang, Jing Shao, Fengwei Yu, and Junjie Yan. Supervision exists everywhere: A data efficient contrastive language-image pre-training paradigm. *arXiv preprint arXiv:2110.05208*, 2021.

[29] I. Y. Liao, M. F. Jelani, Z. Y. Chen, C. K. Wong, and W. C. Wong. Deep convolutional neural networks and their applications to quality classification of germinated oil palm seeds. In *E-Proceedings International Seminar on Digitalisation of Data for Oil Palm Breeding, ISOPB & MBOP*, 2021.

[30] I. Y. Liao, B. S. K. Shen, Z. Y. Chen, M. F. Jelani, C. K. Wong, and W. C. Wong. A preliminary study on germinated oil palm seeds quality classification with convolutional neural networks. In *CVPPA*, 2021.

[31] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv: 2107.13586*, 2021.

[32] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[33] S. Menon and C. Vondrick. Visual classification via description from large language models. *arXiv preprint*, 2022.

[34] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. Language models as knowledge bases? In *EMNLP-IJCNLP*, pages 2463–2473, 2019.

[35] Nina Pörner, Ulli Waltinger, and Hinrich Schütze. BERT is not a knowledge base (yet): Factual knowledge vs. name-based reasoning in unsupervised QA. *arXiv: 1911.03681*, 2019.

[36] Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. Reasoning with language model prompting: A survey. *arXiv preprint arXiv:2212.09597*, 2022.

[37] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[38] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *ACL*, 2016.

[39] Arjuna Chandran Shankar. *86.4% of Malaysia's total licensed oil palm planted area MSPO-certified, says MPOB*. https://theedgemalaysia.com/article/864-malaysias-total-licensed-oil-palm-planted-area-mspocertified-says-mpob, 2021.

[40] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *EMNLP*, pages 4222–4235, 2020.

[41] Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. Flava: A foundational language and vision alignment model. *arXiv preprint arXiv:2112.04482*, 2021.

[42] Richard Socher, Milind Ganjoo, Hamsa Sridhar, Osbert Bastani, Christopher D Manning, and Andrew Y Ng. Zero-shot learning through cross-modal transfer. In *NeurIPS*, 2013.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[44] Xinlong Wang, Xiaosong Zhang, Yue Cao, Wen Wang, Chunhua Shen, and Tiejun Huang. Seggpt: Segmenting everything in context. *arXiv preprint arXiv:2304.03284*, 2023.

[45] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[46] Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*, 2023.

[47] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021.

[48] Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D Manning, and Curtis P Langlotz. Contrastive learning of medical visual representations from paired images and text. *arXiv preprint arXiv:2010.00747*, 2020.

[49] Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations (ICLR 2023)*, 2023.

[50] Zexuan Zhong, Dan Friedman, and Danqi Chen. Factual probing is [MASK]: learning vs. learning to recall. In *NAACL-HLT*, pages 5017–5033, 2021.

[51] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *arXiv: 2109.01134*, 2021.

[52] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. *arXiv: 2203.05557*, 2022.