BOLSTER

Blog / Research Labs
Research Brand Protection

Image Similarity for Brand Detection: A Comparison Between BEiT, SWIN and ViT-MAE

February 22, 2024 | 5 min read

Adithya Singh


Hashing
V/S
BEiT
SWIN
V/S
ViT-MAE

Brand detection is a critical component of Bolster AI, gathered thorough intelligence on malicious phishing and scam entities. The capability to detect the brand of any entity on the internet allows for smart decision making; accurate detection of a brand helps in identifying which brand has potentially been impersonated.

Brand detection can be done in multiple ways: **text classification** of rendered text, **image classification** of the webpage screenshot, **logo detection and classification**. At Bolster AI, we use all these methods to detect the brand of a given webpage to cover maximum known attack vectors expansively.

Brand detection through image classification is one of the most potent variables in detecting phishing scams. If two webpages have same detected brand, image classification can be done in multiple ways. Some of these methods have been discussed in **our recent computer vision blog**.

Diving deep into the process of image classification, we find two possible ways to solve the problem of brand detection. A popular method is image classification with a **neural network**.

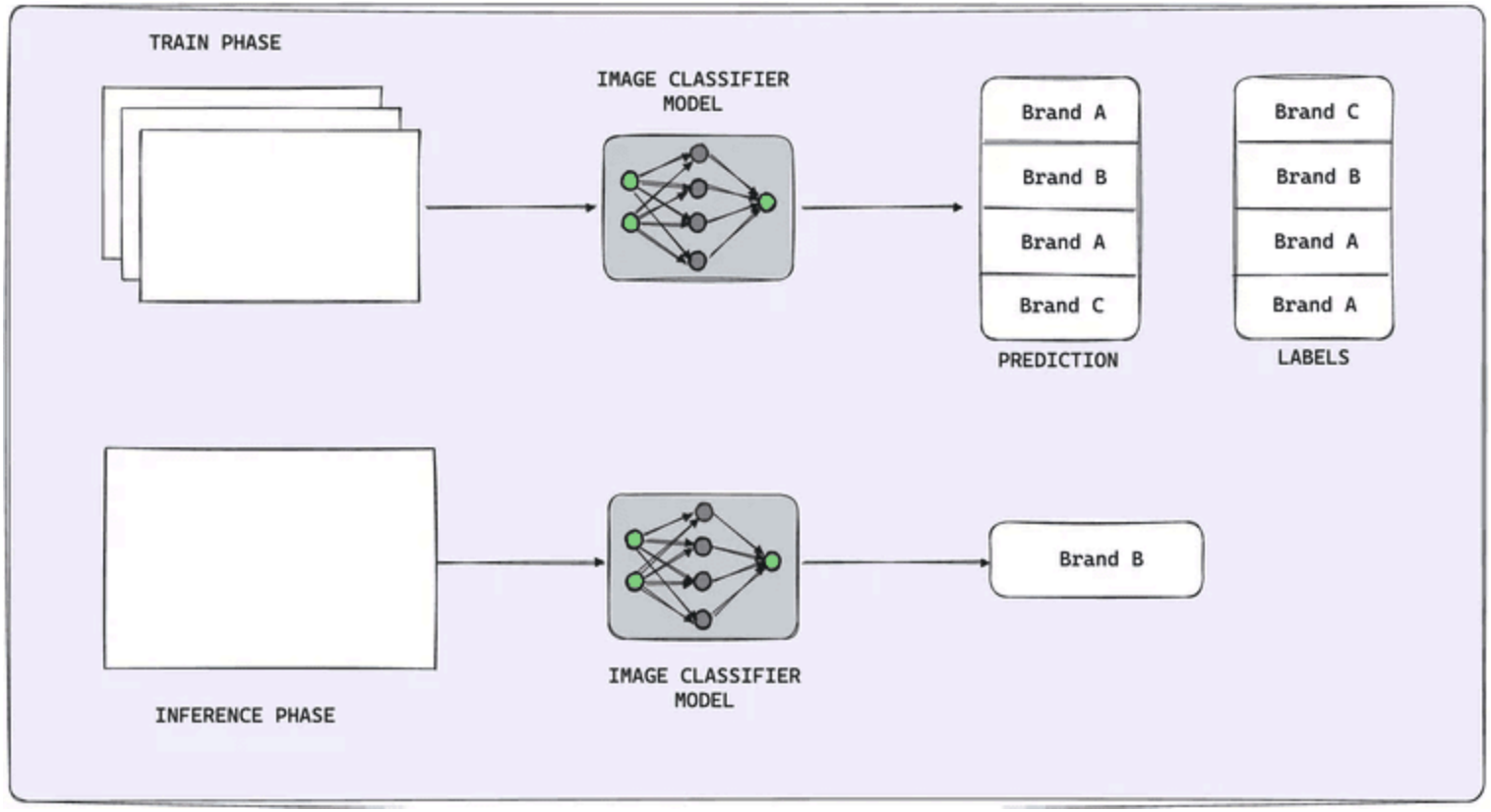

Figure 1: Image classification with CNNs

Convolutional neural networks (CNNs) are largely popular in solving computer vision problems like image classification.

The idea is to gather a large dataset of images. In the context of brand detection, these images are the screenshots of webpages. Each image is labeled with the brand name and fed as examples to a CNN.

The network is trained over many iterations to make predictions for brand labels using the backpropagation with the objective to minimize a cross entropy loss function. The popularity of CNNs is backed by the efficacy of the method; the classification is reliable and generalizes well.

However, the performance of a CNN is guaranteed only with an expansive, well-labeled and well-represented dataset. The CNNs need a balanced distribution among examples for all the classes.

The method struggles in the case of extreme imbalances with the dataset. For the application in the context of brand detection, this is precisely the biggest roadblock in developing a functional system. A dataset can suffer from a severe imbalance, where some brands have at most one example, while others have thousands.

This roadblock gives rise to the need of developing an algorithm capable of being unaffected by severe imbalances of data. This is where classification through image similarity comes in. Let's dive deep into the idea of image similarity and the possible approaches.

Brand Detection using Image Similarity

Image similarity relies on the idea of finding similar matches for images. The assumption is that most similar looking webpages belong to the same brand or to an entity impersonating some genuine brand.

There are multiple ways to implement image similarity retrieval system. The first approach is via hashing method; another approach is with embedding similarity search.

Both methods however, follow same principle of obtaining a lower dimensional representation of an image and matching this representation to the data already possessed. The approaches are different on the basis of how they obtain the mentioned latent representation and have their own advantages and disadvantages.

We will go deep into both methods and their sub approaches.

Figure 2: Image classification with similarity search

The question remains about how image similarity is a better alternative to image classification for the given problem. All similarity matching methods require collection of a dataset, which for this problem is a collection of images, these images are the screenshots of webpages.

Each screenshot is given a label corresponding to which brand or organization the webpage belongs to.

An inherent obstacle is the severe imbalance in the collected dataset. Some brands have multiple designs and diverse webpages, while some brands own a single webpage website. This ends up creating an imbalance where some brands have as low as one labeled webpage belonging to them, while some other brands have around thousands of different webpages belonging to it.

This creates a heavy skew in the dataset, and traditional image classification through CNNs struggles to generalize over the problem.

Image similarity search handles this problem easily, as it tries to find the best match for an input image and thus it is independent of the data distribution.

Even if a particular brand has only a single image in the stored dataset, if it is the best match for the query image the system solves the problem reliably.

There are other advantages to image similarity method:

No training required: The technique uses the off-the-shelf method to convert images to latent representations, hence there is no requirement to keep running the training cycles. This is a considerable optimization on compute resources.

Low computation: The searching for best match of the input image is a computationally efficient process, since the method relies on latent representations only. The representations are quick to search for and can be heavily optimized. The hashing conducts search in constant time, whereas searching via embedding is a linear search.

Analyze the Different Approaches

There are multiple approaches for conducting image similarity search. All methods rely on converting the image to latent representation. The difference is in terms of the type of representation obtained. Primarily, our systems conduct similarity searches in two ways: hashing, and transformer based embedding matching.

Computer vision hashing

This hashing method is a straightforward method to implement. The idea is to obtain a hexadecimal representation of an input image. This done using a Secure Hashing Algorithm or SHA.

This method returns a 64 bit hexadecimal encoding of any input value. For our problem we use it for encoding the images. The encoded representation is unique to the image.

Once obtained, the hash encoding can be stored in a look up table or a database.

For this problem, an expansive database is created by converting thousands of webpages belonging to about 1000 brands and organizations. Storing and searching through this database is a constant time operation.

Whenever a query image is now received, it is also converted to this encoded hexadecimal hash, this hash is then looked up in the database.

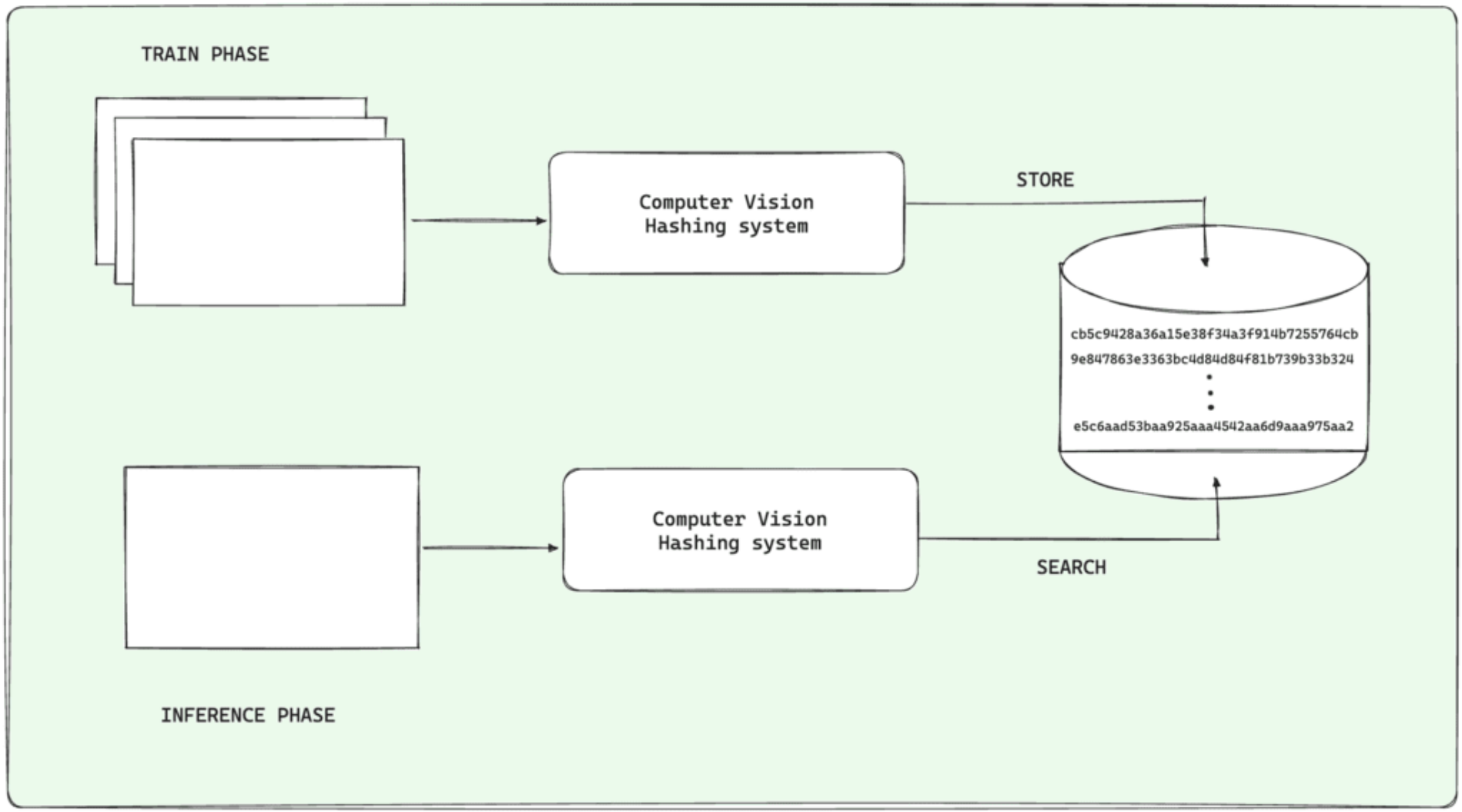

Figure 3: Image classification with a hashing system

The constant time storing and searching is only possible due to the fact that each image has a unique hash representation, and hash tables are extensively optimized to handle them.

Few things to note here however; the hexadecimal encoding generated by the hashing system is not a meaningful representation of the image. This implies that similar images will not have similar hashes, and if one tries to project the hash encoding to a latent space, similar images will not lie together.

A positive result is generated only when an exact match is found. Even if two images are very closely similar but slightly different, the match result returned will be negative as they will have completely different hash representations.

Embedding based similarity search

Embedding based methods try to overcome the shortcoming of hashing. The principle is the same in the sense that, images are converted to a lower dimensional latent representation. These representations are then compared to find the best batch.

As discussed above, representations obtained through hashing methods are not meaningful. This problem is resolved by embedding based methods by generating embedding that are encodings of the image features.

This means that embedded encodings of images with similar features are also similar. More mathematically, if embedding encoding of images are projected on latent space, then the vector of similar images lie closer to each other.

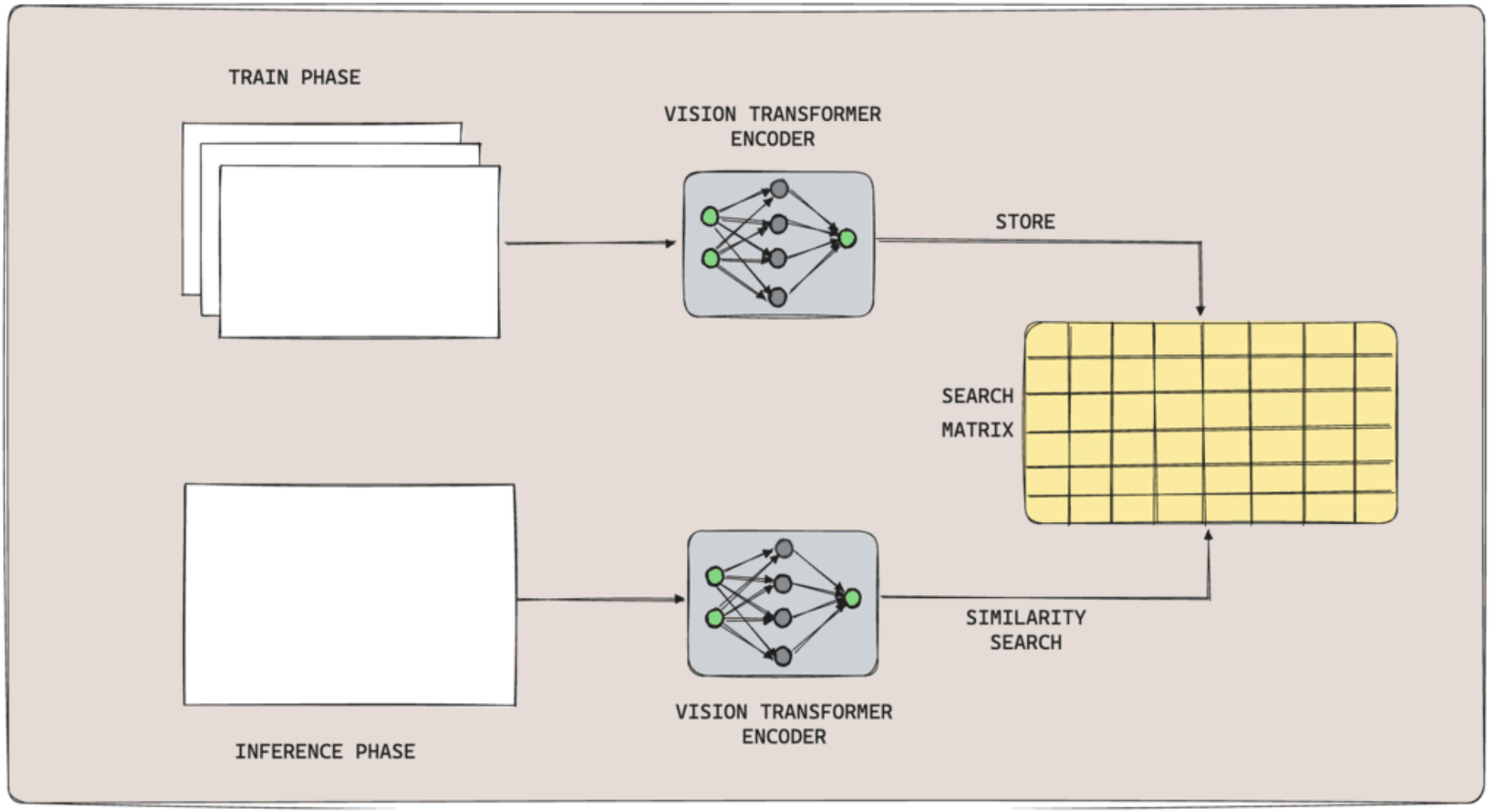

Figure 4 : Embedding based image similarity search

A match is found even if an image is not exactly same to an image in database, but is a very similar. This allows for a more generalized system capable of detecting the best brand match of a given query image from the existing database.

For generating embeddings, we deploy neural networks based on transformer architectures.

For a long time, CNNs have been de facto models to solve image classification problems. The rise of transformers however have drawn a serious attention of computer vision researchers especially after their groundbreaking success in NLP problems.

Vision transformers have been able to bridge the gap between transformers for NLP and computer vision. Transformers are designed on same principles as autoencoder models, hence they are excellent at extracting the embedding encoding from the images.

We tried out 3 different vision transformer approaches to realize the embedding based similarity search systems.

1. BERT Pre-training of image transformers (BEiT)

This model is designed on a very simple principle of pre-training a model to solve the problem of annotated data scarcity for training vision transformers.

This paper solves this problem by introducing the self supervised training paradigm for vision transformer. The idea is to train a vision transformer for a task that does not require labeled images. Computer vision is often bottlenecked by the requirement of annotated images.

The annotation of image is a significant burden of effort and time especially for use cases like object detection and image segmentation. This problem can be solved by using self supervised learning based training with a pre-training task which doesn't require labels to train, and still allows the model to learn deep patterns in the images.

For BEiT, this pre-training task is masked image modeling.

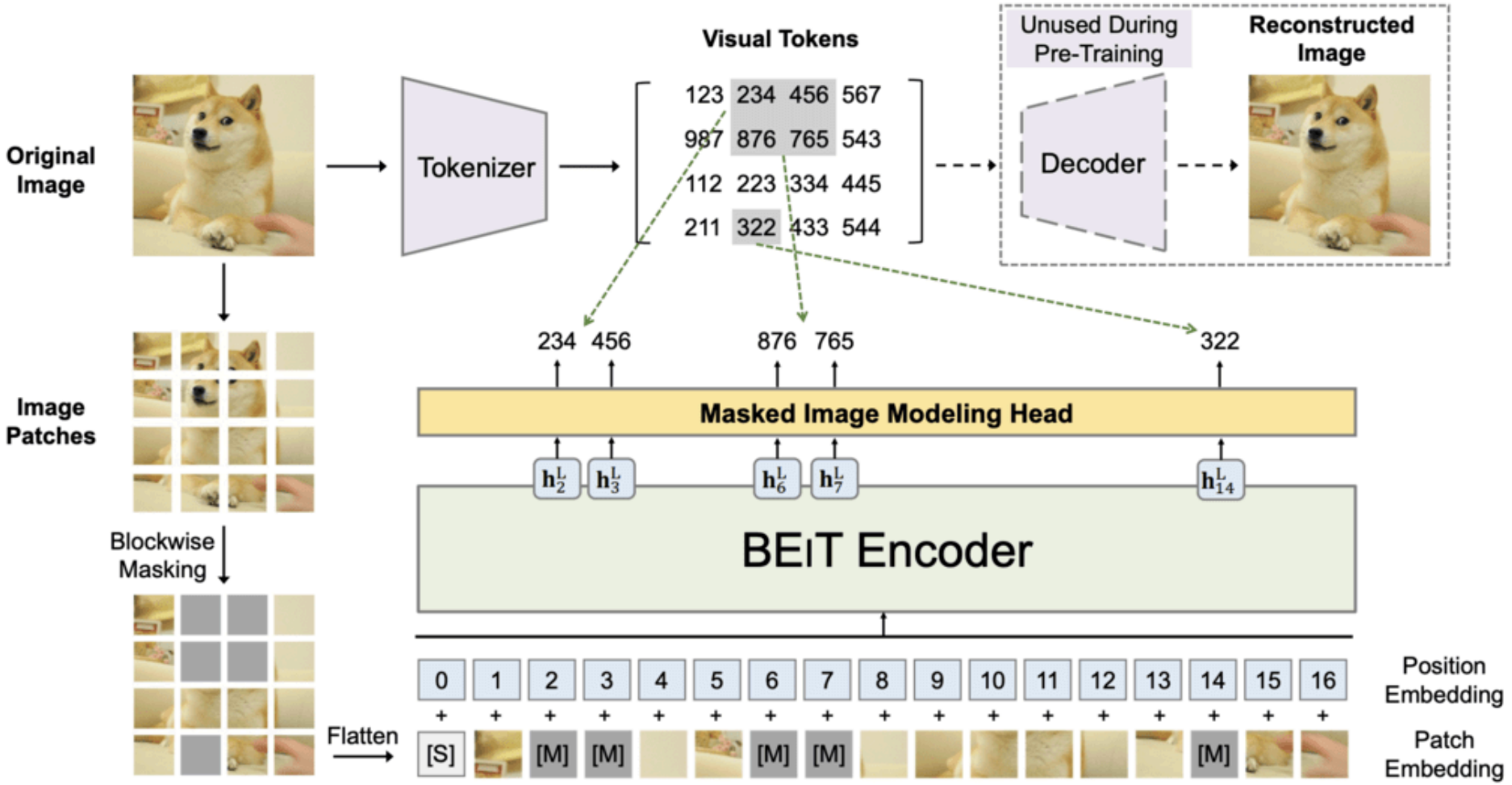

Figure 5: BEiT architecture

Masked image modeling is the task to train model to reconstruct masked patches in the input images. The vanilla vision transformer breaks the input image in smaller patches to form a sequence.

BEiT makes a small modification to this step and actually masks a portion of these patches. Then the vision transformer encoder makes a prediction to encode all the patches including the masked ones.

This encoding is then evaluated for error against encodings generated by a separate image tokenizer.

The image tokenizer is an off the shelf trained token generator for input image patches. The error computed for the encodings generated by the vision transformer encoder is then used to adjust the weights of the encoder during back propagation. This allows network to get a rich understanding of underlying image patterns, and all this is done without the requirements of explicit labels.

This implies for this training vast expanses of unlabeled image data available on the internet can be utilized. Once pre-trained, the model can be connected to a decoder to train on downstream tasks like object detection and image segmentation.

For our use case, we use the encoder module of BEiT for generating embedded encodings of the webpage screenshot, which can then be used to search for the closest match in the prior-collected database.

For a deep dive into BEiT architecture, **check out our previous post**.

2. Vision Transformer based on Masked Autoencoding (ViT-MAE):

ViT MAE model was proposed with the motivation of presenting vision transformers as scalable learners. A significant damper on scaling vision transformers is the requirement of huge datasets to achieve the similar level of performances which CNNs can achieve with relatively much smaller datasets.

This poses a scalability problem, as its very uncommon for vision tasks to use huge dataset. The collection and labeling of image datasets is a serious overload on effort and time resources. **ViT-MAE** solves this problem by training vision transformers as autoencoders on masked image modeling task using self-supervised learning paradigm.

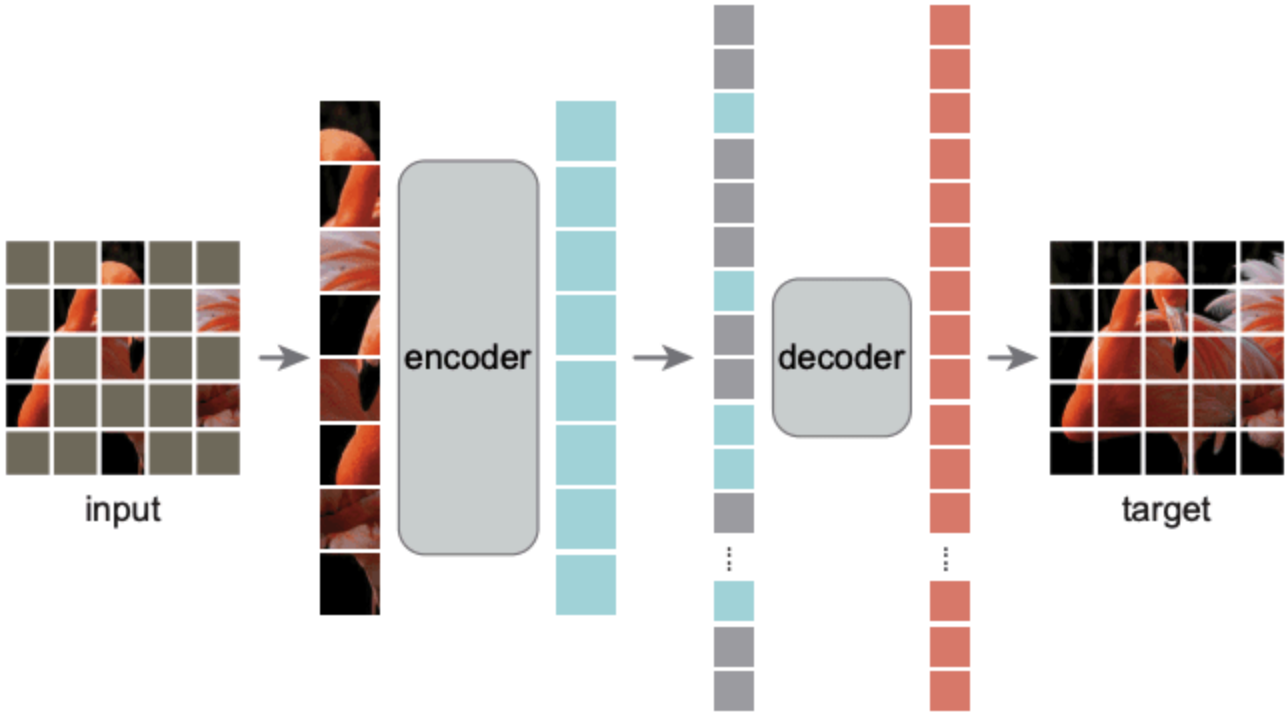

Figure 6: ViT-MAE architecture

In the input image, 75% patches are randomly masked; the encoder module of ViT only takes unmasked patches as input, and produces an embedding. This embedding is then concatenated with learnable masked image patch encoding.

The decoder then uses this concatenated embedding sequence to recreate the raw image pixels of the masked patches of the input images. The self-supervised training allows for unlabeled dataset to be used, which alleviates the problem of requiring huge datasets, and makes vision transformers scalable.

For our use case, we use the encoder module of ViT-MAE for generating embedded encodings of the webpage screenshot, which can then be used to search for the closest match in the prior-collected database. For a deep dive into ViT-MAE architecture, **check out our previous post**.

3. SWIN Transformer

SWIN Transformer model was introduced with the motivation to draw the bridge between CNNs and Vision Transformers. The authors argue that there are key differences between language and vision data is of the variation in scale between image features and language tokens.

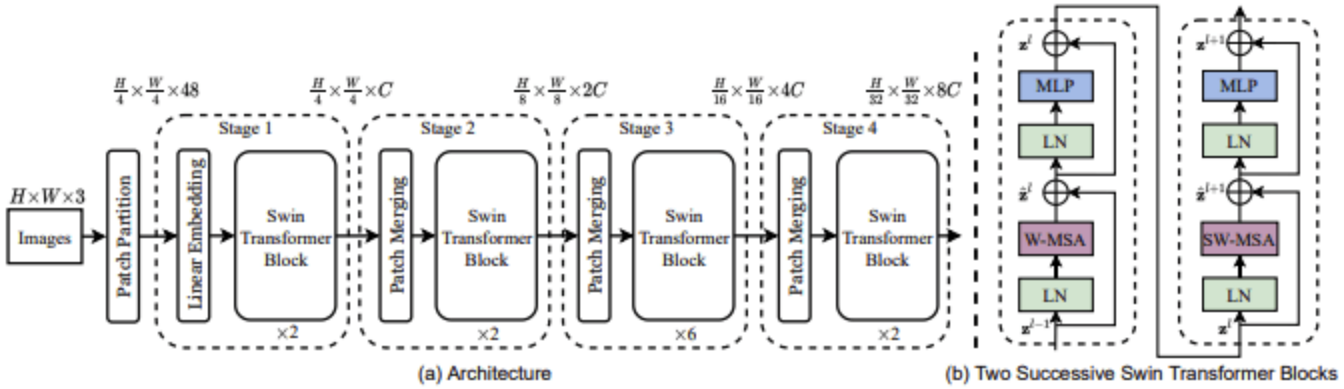

Figure 7 : SWIN architecture

SWIN is a hierarchical transformer which addresses this problem of scale variation by computing transformer representation with shifted windows. The idea is to further divide usual image patches of input image to even smaller patches. These smaller non overlapping patches are then presented to attention layers.

The output from these attention layers are then concatenated in pairs to combine attention output the two higher level patches, this concatenated output is presented to next set of attention modules.

This hierarchical propagation through attention layers, allows transformer to pay attention to smaller scale features and deal with variation in scales for image data. For a deep dive into SWIN architecture, **check out our previous post**.

Results and Observations

Now that we have a fair understanding of all the models to use similarity search methods including the hashing and embedding based system, it's time to conduct experiments and find the best performing method.

For this we collected a dataset of about 45 thousand webpage screenshots, distributed among close to 1000 brands and organizations. For each of the four approaches, encodings are extracted and stored.

For hashing system, these encodings are hexadecimal and stores as a database, for all three embedding based methods the encodings are stored in a search matrix, where each column represents embedding of a single webpage.

We also prepared a test set of **10000 unseen webpage screenshots**, to evaluate the performance of the network. For the screenshots in test set, each model makes a prediction.

We compare the models on the basis of **precision**, **recall** and an aggregate metric of **F1 score**.

Computer vision hashing results

Brand	Precision	Recall	F1 Score
Brand A	1	0.6496	0.7875
Brand B	1	0.9642	0.9818
Brand C	1	0.9375	0.9677
Brand D	1	1	1
Brand E	1	0.7929	0.8845
Brand F	1	1	1
Brand G	1	0.9	0.9473
Brand H	1	0.8888	0.9411
Brand I	1	0.5	0.6666
Brand J	1	0.2	0.3333

BEiT model results

Brand	Precision	Recall	F1 Score
Brand A	1	0.6496	0.7875
Brand B	1	0.9642	0.9818
Brand C	1	0.9375	0.9677
Brand D	1	1	1
Brand E	1	0.7929	0.8845
Brand F	1	1	1
Brand G	1	0.9	0.9473
Brand H	1	0.8888	0.9411
Brand I	1	0.5	0.6666
Brand J	1	0.2	0.3333

SWIN model results

Brand	Precision	Recall	F1 Score
Brand A	1	0.6496	0.7875
Brand B	1	0.9642	0.9818
Brand C	1	0.9375	0.9677
Brand D	1	1	1
Brand E	1	0.7929	0.8845
Brand F	1	1	1
Brand G	1	0.9	0.9473
Brand H	1	0.8888	0.9411
Brand I	1	0.5	0.6666
Brand J	1	0.2	0.3333

ViT-MAE model results

Brand	Precision	Recall	F1 Score
Brand A	1	0.6496	0.7875
Brand B	1	0.9642	0.9818
Brand C	1	0.9375	0.9677
Brand D	1	1	1
Brand E	1	0.7929	0.8845
Brand F	1	1	1
Brand G	1	0.9	0.9473
Brand H	1	0.8888	0.9411
Brand I	1	0.5	0.6666
Brand J	1	0.2	0.3333

The results illustrated above are for 10 most popular brands that we collected data for. From primary analysis, it can already be seen that SWIN transformer is performing better than other methods, and is a significant improvement over the existing hashing system.

Full fledged experiments portray an even clearer picture of superior performance os SWIN model, where we evaluate the performance of the models on close to 1000 brands. The SWIN model detects **54.55% more accurate predictions** when compared to the hashing systems.

It can be seen that all three vision transformer models perform significantly better than hashing systems. This is due to the capability of vision transformer models to capture a meaningful latent representation of the images unlike hashing systems which rely on one to one match.

In other words, hashing systems are excellent at tackling false positives but have high count of false negatives as even a slight change in an image results in completely different hash. This is also apparent from precision metric of hashing system, a perfect score.

Vision transformers rely on capturing the features of the images and similar images produce similar embedding, which syncs well with the assumption that similar webpages belong to the same website or is a case of brand impersonation.

In our workflow, we use a combination of hashing systems and a SWIN vision transformer to tackle both problems regarding false positives and false negatives to boost our overall brand detection capabilities.

Conclusion: Protecting Your Brand

Detecting brand attacks with a tool powered by AI is a potent weapon in the defense against phishing attacks. With the capability of detecting the brand associated with a webpage, it becomes possible to detect brand impersonation at scale.

If a webpage is detected to be belonging to a brand but is not officially hosted by a brand, it is highly probable to be a phishing attack.

There are multiple ways to perform brand detection, one of these ways is using the computer vision techniques by analyzing the screenshots of the webpage.

Detection tasks in computer vision are usually carried out by CNN based image classifiers. Classification using CNNs has its merit, but exhibits weak performance when datasets are severely imbalanced.

For brand detection, some brands may have only one webpage associated to them, while others might have 100 webpages hosted on their domain. CNNs struggle with misrepresented classes and tends to be biased.

This problem is solved by implementing brand classification with image similarity. With image similarity, the objective is to convert images into a lower dimensional latent representation or encoding, storing these encoding into a database, when a new image is received, it is also converted to an encoding and this encoding is searched for a match in the existing database.

There are multiple methodologies to obtain the encoding. Hashing systems generate a hexadecimal encoding of the image using Secure Hashing Algorithm.

Vision transformers generate embedding vectors as encoding. While hashing systems provide a highly accurate system, but the number of detections are low because system does not generalize well and provides a match only when the webpage is already in the database.

Vision transformers have a higher generalizing capability, and can get accurate matches even if they have not seen the exact webpage before.

We conducted extensive experiments on different vision transformers like BEiT, SWIN and ViT-MAE and compared their efficacy to one another and the existing hashing system. Vision transformers exhibit a significantly stronger performance when evaluated on same data against the hashing system.

Among the three vision transformers, **SWIN exhibits the best performance.**

At Bolster, we have deployed an ensemble of hashing and SWIN transformer image similarity system to conduct a robust brand detection on thousands of webpages sourced from the internet. This helps us cover the attack surface of brand impersonation extensively, bringing down both false positives and false negatives in our detection.

For inside access to Bolster's use of SWIN transformers, and to see how we can protect your brand from scam sites and phishing attacks, **request a demo with us today.**

References for this research blog:

1. BEiT paper : **BEiT : BERT Pre-training for image transformers**
2. BEiT blog : **BERT for Image Transformers (BEiT) : A Definitive Guide to Computer Vision Breakthrough**
3. SWIN paper : **Swin Transformer: Hierarchical Vision Transformer using Shifted Windows**
4. SWIN blog : **SWIN Transformers: The Best of Two Worlds**
5. ViT-MAE paper : **Masked Autoencoders Are Scalable Vision Learners**
6. ViT-MAE blog : **VIT-MAE: Scalable Learning for Vision Transformers**