Analysis for 2021 Kaggle "Shopee-Price Match Guarantee" Competition

Pu Tan*

Abstract : With the development of computer vision, assistance from different modalities is needed to accomplish more challenging tasks. To improve product matching, both visual and textual information of products can be utilized for complementarity. Effective fusion of cross-modalities has become the key to task completion. This paper aims to recap the solution for the 2021 Kaggle "Shopee-Price Match Guarantee" competition [2]. The methods will be elaborated from the perspectives of image, text and post-processing, along with experimental results on public and private datasets. Our final solution achieved F1 Score of 0.756 on the public leaderboard (ranked 36/2426, top 2%) and 0.744 on the private leaderboard (ranked 39/2426, top 2%).

Key words: E-commerce, product matching, machine learning, deep learning, multi-modality; Image retrieval; text embedding

1 Introduction

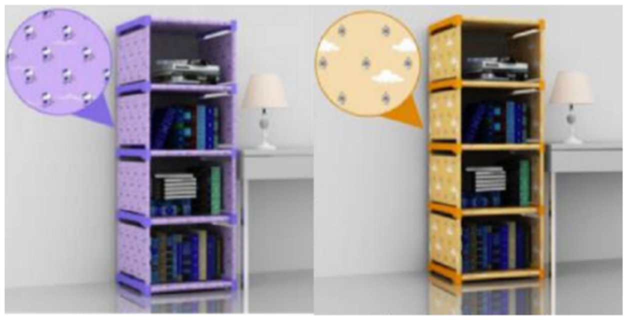

Figure1. Similar Images of Different Products

When shopping, consumers often compare prices of the same product from different sellers, hoping to get their desired items with the lowest cost. Retailers also want to provide such comparison reports to ensure consumers that their products are the most cost-effective, hence improving competitiveness. Image matching is commonly used for this task. However, due to variations in color, size, shooting angles or label noise, the same product may have very different visual features. Similarly, different products may share similar image representations (Figure 1). Retailers want to avoid false claims and improper promotions by mixing up different products. Therefore, to improve matching accuracy, it is popular to combine deep learning and traditional machine learning methods to analyze both visual and textual information for comparing product similarity.

In recent years, rapid progress has been made in deep learning for computer vision and natural language processing. There is increasing interest in combining them to accomplish more complex and diverse tasks. Multi-modality has become a hot topic in AI research. Generally, it refers to processing and fusing different data types like images, videos, texts and audios to fulfill higher requirements. In this competition, mainly the visual information from product cover images and textual information from product titles on shopping websites are utilized to find out the matching product labels for each sample.

The main methods used in this paper include:

1) Image module: Use NFNet and EfficientNet as backbone models to extract image features of products.
2) Text module: Use Bert as backbone model, supplemented by TFIDF to extract text features of products.
3) Post-processing: Use KNN to calculate cosine similarity between features, and query expansion to improve recall. Take the union of image and text retrieval results, and gradually adjust the distance threshold via dynamic thresholding to ensure retrieving at least one result.

2 Related Work

2.1 Image Recognition and Retrieval

The rapid development of deep learning in image recognition was set into motion in 2012 with the introduction of AlexNet, which reduced the error rate of ImageNet classification by nearly 10% [8, 14]. In the following years, various CNN architectures were proposed. Outstanding models like GoogLeNet and VGGNet significantly shone in the 2014 ImageNet competition [19, 16], prompting researchers to realize the necessity of deeper layers in CNNs for improved performance.

In 2015, Kaiming He and colleagues proposed a Deep Residual Network (ResNet) [7] to address the optimization issues of deep CNNs, achieving image recognition accuracy that surpasses human performance. To this day, ResNet serves as a powerful backbone that is widely applied. Researchers, having improved accuracy, shifted their focus on reducing model size and boosting inference speed. Google introduced EfficientNet in 2019 [20], which explored the depth, width, and image resolution of the network to balance speed and accuracy. Further, in 2021, Google built upon ResNet to propose the NFNet [4], which outperforms EfficientNet in both speed and accuracy by removing Batch Normalization and employing adaptive gradient clipping.

These models, based on image recognition, have been extended to other visual tasks such as object detection, semantic segmentation, and image retrieval. Image retrieval is most closely related to the task discussed in this paper. In a typical image retrieval dataset, there are two parts: a Query set and a Gallery set. Algorithms aim to match each image in the Query set to the images in the Gallery set that belong to the same class. The effectiveness is usually evaluated based on a trade-off between precision and recall. The visual part of this competition also falls under the category of image retrieval. However, the dataset is not divided into Query and Gallery; instead, it retrieves tags belonging to the same product from a single repository based on image similarity.

Inspired by image retrieval tasks, some simple yet effective techniques can also be applied to this competition, such as Query Expansion [3, 11], which improves the recall rate of retrieval by averaging or weighted summing the features of the top-k matched images.

2.2 Text Feature Extraction

To achieve more effective product matching, in addition to image information, text information also plays an auxiliary role. In measuring text similarity, a variety of methods can be applied. Generally, the frequency of word occurrence in the text can be counted, i.e., the Term Frequency (TF) method [12], which reflects the importance of a word in a text. However, the text often contains a large number of stopwords that have no actual meaning; therefore, counting word frequency alone cannot adequately reflect text features. This led to the introduction of the Inverse Document Frequency (IDF) method, which assigns an importance weight to each word based on its frequency. Combining the two methods gives rise to the TF-IDF approach.

With the advent of deep learning, some excellent methods have also emerged in the field of Natural Language Processing (NLP). In 2017, Google's translation team completely abandoned CNN and RNN structures, introducing a Transformer model based on the Self-Attention mechanism [21] to solve text translation issues. Over the subsequent years, due to its excellent properties like dynamic weight allocation and global relationship modeling, Transformers have completely revolutionized the NLP field, making breakthroughs in almost all tasks. Specifically, in 2018, Google further proposed Bert, a pre-trained language model based on bidirectional encoding built upon the Transformer architecture [6]. Bert achieved state-of-the-art (SOTA) results in 11 NLP downstream tasks and has a milestone significance in the history of NLP development. In 2019, to address Bert's limitations in measuring semantic similarity directly, Nils Reimers et al. introduced Sentence-Bert, a Siamese Bert network [13], reducing inference time in text matching tasks from 65 hours to 5 seconds.

In this paper, we use both Tfidf and Sbert methods for text feature extraction and similarity measurement.

2.3 Loss Functions in Similarity Measurement

The competition task closely aligns with the domains of Face Recognition and Person Re-identification. In these realms, a plethora of effective loss functions have been devised to improve similarity measurement. Contrastive Loss [17] is commonly employed for the training of Siamese networks. By taking in two images as input for each training cycle, it more accurately represents the degree of matching between paired samples. In FaceNet [15], the Triplet Loss function was introduced. Within a given triplet input, an anchor is associated with a positive and a negative sample to form positive-negative sample pairs, thereby discriminating similar samples effectively. Subsequent improvements have been made to Triplet Loss as well.

These aforementioned loss functions fall under the category of metric learning, aiming to optimize the network by computing the distances between features. A closely related approach, representation learning, treats the task of similarity as a classification problem. Modifications to the Softmax Loss function have been made to minimize within-class variance and maximize between-class variance. For example, Center Loss [23] constrains features within the same class to be closer to their center point. In SphereFace [9], CosFace [22], and ArcFace [5], the concept of Margin penalty has been adopted to increase the training difficulty, consequently enhancing the discriminative power of the features. Circle Loss [18] further unifies metric loss with classification loss to formulate a more generalized loss function.

3 Method

3.1 Image Feature Extraction

To learn discriminative features, this study employs the loss function from ArcFace during the fine-tuning stage for classifying the dataset. However, the test set contains an unknown number of product categories that differ from those in the training set, rendering it unsuitable for direct classification tasks. Consequently, during inference, we only extract features from the final fully connected (FC) layer. These features are then used to measure the distances between different image attributes, and products belonging to the same category are grouped together based on these distances.

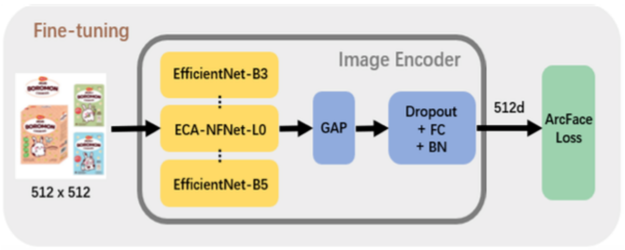

Figure 2. Main Workflow of the Image Fine-Tuning Section
(Image first goes through encoder to obtain a 512-dimensional feature vector, ArcFace Loss is employed for loss computation)

Fine-Tuning:

The main workflow for fine-tuning is as illustrated in Figure 2. Initially, an image is fed into an image encoder to obtain a 512-dimensional image embedding. The embedding feature is then used to compute loss via ArcFace Loss[5]. Within the image encoder, the image first passes through the main body of the model (either EfficientNet-B3, EfficientNet-B5, or ECA-NfNet-L0). Subsequently, the resultant feature map undergoes Global Average Pooling (GAP), followed by Dropout, a Fully Connected layer (FC), and Batch Normalization (BN) to reduce the feature dimensionality to 512. Specific fine-tuning settings are discussed in Section 4.2.

Inference:

As shown in Figure 3, during the inference stage, the input image is first resized to 512x512 dimensions and fed into the trained image encoders, obtaining three separate 512-dimensional image features. To concatenate these three embeddings, each is subjected to L2 normalization and then multiplied by corresponding weights. The purpose of this operation is to give greater emphasis to single models that perform better. The concatenated output yields a 1536-dimensional feature vector, serving as the final image embedding for subsequent similarity calculations.

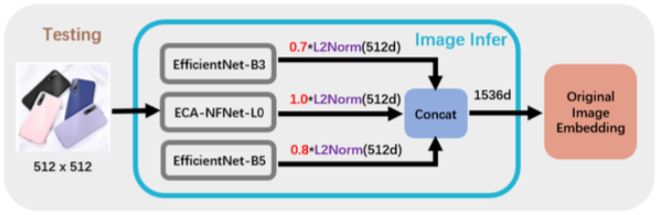

Figure 3. Main Workflow of the Image Inference Stage
(Test images pass through separate models for feature extraction. Extracted features, after being weighted, are concatenated to serve as the final output)

3.2 Text Feature Extraction

Text processing primarily involves two components: Tfidf and SBert. Tfidf does not require training, whereas Bert models need to be fine-tuned on the Shopee dataset. The model selection for Bert can vary; for instance, Indonesian DistilBERT can be used for fine-tuning given that the dataset includes a substantial amount of text in the Indonesian language. In this study, we ultimately chose Sbert (paraphrase-xlm-r-multilingual-v1) for feature extraction.

Fine-tuning Process:

As illustrated in Figure 4, the input for fine-tuning consists of product text descriptions from the Shopee dataset, which are fed into the pre-trained SBert model utilizing a Siamese network architecture. For classification purposes, only the first dimension (i.e., [CLS] token) of the model output is used, yielding a 768-dimensional feature vector. Similar to the image model, ArcMarginLoss is employed for fine-tuning the SBert model to extract distinct features.

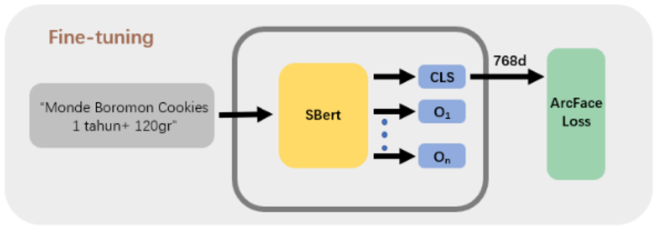

Figure 4. Main Workflow of the Text Fine-Tuning
(Pre-trained SBert model process training text to extract features, outputs go through CLS to ArcFaceLoss to optimize the feature representation)

Inference:

As illustrated in Figure 5, during the inference stage, the input text is processed through both the SBert and Tfidf models. These models produce feature vectors of dimensions 768 and 24,939, respectively, which are then employed for similarity calculations as will be elaborated upon in Section 3.2. It should be noted that the dimensionality of the features extracted via Tfidf can be manipulated by adjusting the max_feature parameter.

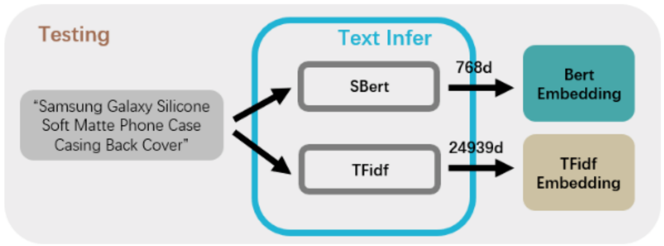

Figure 5. Main Workflow of Text Inference
(Test text as input, undergoes feature extraction by both the SBert and TFIDF models. Feature vectors are then used for subsequent similarity calculations)

3.3 Post- processing

The main procedure for post-processing is illustrated by Figure.6. The post-processing stage plays a significant role in enhancing the model's performance, in addition to the feature extraction methods described in sections 3.1 and 3.2 and has been identified as a critical factor for boosting performance.

Initially, the processed dataset will consist of N Image Embeddings, Tfidf Embeddings, and Bert Embeddings, where N is the size of the dataset. To improve the model's recall, we employed the αQE (Alpha Query Expansion) method [11]. This approach is found to adapt better to various data distributions compared to AQE (Average Query Expansion) [3].

The formulas for AQE and αQE are respectively shown in Equations 1 and 2. Here $f(q)$ represents the features of a sample q, and $f_q(top_i)$ signifies the features that have the ith smallest cosine distance to sample q. The core idea of αQE is to use the distance raised to the power of α as a weight for summing up the top $k$similar features. This weighted sum leads to tighter connections between the processed sample and its k most similar samples, thus enhancing recall in subsequent steps. When $\alpha = 0$, αQE reduces to AQE.

$$AQE\colon f(q) = \frac{1}{n}\sum_{i=1}^{n} f_q(top_i) \qquad (1)$$

$$\alpha QE\colon f(q) = \frac{1}{n}\sum_{i=1}^{n}\left[f(q)^T f_q(top_i)\right]^{\alpha} f_q(top_i) \qquad (2)$$

Before feature fusion, the K-nearest neighbors (KNN) are identified for each feature based on cosine distance, and they are sorted according to this distance. Due to the varying scales of overall similarity in different feature spaces, individual thresholds are set for Image, Bert, and Tfidf features. For each sample, matches with distances smaller than the threshold are included in the sample's group, effectively completing the prediction.

However, experimental analysis reveals that a significant number of groups in the training set have only one match, impacting the model's recall negatively. Simple thresholding also results in thousands of singleton groups in the predictions (i.e., groups where no other samples match apart from the sample itself). To address this issue, a "Dynamic Thresholding" strategy was introduced to significantly enhance the model's recall.

Dynamic thresholding uses a lower limit L, an upper limit H , and a step size S to adjust the threshold iteratively. The loop exits when a sample obtains at least one match (i.e., the size of the match set is greater than or equal to 2) at a particular threshold. To further balance precision and recall, the union of matches from Image, Bert, and Tfidf features is considered as the final result during each thresholding iteration. This allows the algorithm to exit the loop if any of the three features find a match, mitigating the risk of introducing erroneous labels due to lenient conditions.

Experimental results indicate that the initial threshold should be slightly lower than the optimal threshold of individual models, while the impact of the upper limit of the threshold on the final outcome is minimal. Specific settings are discussed in Section 4.2.

4 Experiments

4.1 Dataset

The Shopee dataset is divided into a training set and a test set. The training set consists of 34,250 samples and includes 11,014 types of products. Each sample contains the product code, image, image hash value, product title, and the category to which the product belongs. Analysis shows that the largest class in the training set contains 50 samples, while the smallest class contains just 2 samples, accounting for approximately 20% of the total samples. The test set includes over 70,000 samples, but the specific details are not publicly available.

The preliminary leaderboard ("A-board") score is calculated based on 40% of the test set, and this score becomes visible to participants upon submission. The final score, which determines the final ranking, is generated from the remaining 60% of the test samples and is disclosed only after the competition ends.

The objective of the competition is to categorize products correctly based on the provided image and title information. The performance metric used for evaluation is the F1 Score, which balances precision and recall, as indicated in Equation 3 (not shown here).

$$F_1 = 2\,\frac{precision \times recall}{precision + recall} \qquad (3)$$

The dataset presents various challenges, both in the image and text domains. For example, as shown in Figure 1 (not displayed here), items that belong to different categories may only differ in color. Additionally, there are instances where different categories of products have almost identical text titles. These factors require participants to make comprehensive judgments by combining both image and text information.

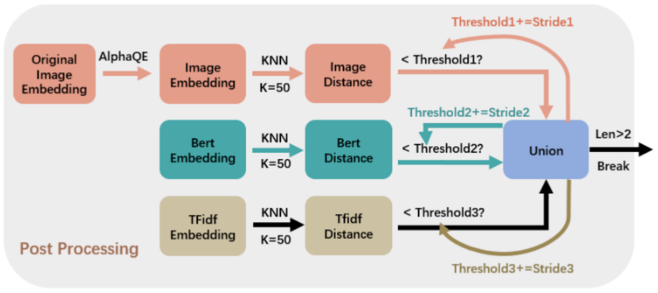

Figure 6. Workflow of post-processing
(AlphaQE is applied to embedded image features, then KNN calculates cosine distances for each feature. Thresholds are dynamically adjusted until a match is found)

4.2 Experimental Setup

Image Part: Three separate models, EfficientNet-B3, EfficientNet-B5, and ECA-NFNet-L0, are fine-tuned individually. Before fine-tuning, all models are initialized with pre-trained parameters from ImageNet. Images are resized to 512x512 and undergo data augmentation including horizontal and vertical flips, random rotations, and brightness variations. The ArcMarginLoss loss function is used for classification across 11,014 categories. Scale and Margin parameters are set at 30 and 0.5 respectively. Training is done over 15 epochs with a batch size of 8 using the Ranger (RAdam+LookAhead) optimizer. For inference, the weights for the three models are set to 0.7, 0.8, and 1.0 based on their individual accuracies.

Text Part: Fine-tuning is done on top of pre-trained Sbert(paraphrase-xlm-r-multilingual-v1) model. ArcMarginLoss is also used for multi-class classification over 11,014 categories. The model is trained for 30 epochs with a batch size of 32 using the AdamW optimizer. The first two epochs use a warm-up strategy with an initial learning rate of 5e-5. Unlike the image models, the output dimension is retained at 768. During inference, Bert's batch size is set to 16 and Tfidf uses 'English' for stop words and sets max_feature to 25,000.

Post-Processing: KNN metric is set to "cosine" for cosine similarity. Up to 50 nearest neighbors are considered for each sample in the test set. AlphaQE is used to improve retrieval recall. Topk is set to 2 for exponential weighted averaging with Alpha=3.0 and Count=1. Dynamic thresholds are used for filtering similarity scores from image, Tfidf, and Bert features, with limits and steps detailed in Table 1.

	Low(L)	High(H)	Stride(S)
Image	0.24	0.50	0.01
TFIDF	0.25	0.50	0.20
Bert	0.08	0.30	0.01

Table1. Dynamic Threshold Parameters

4.3 Experimental Result

This section will present the experimental results, all of which are measured by the F1-Score. Ablation studies have been conducted on single CV (Cross-Validation) models, multiple CV models, DT (Dynamic Thresholding), and QE (Query Expansion). The term "Public" refers to 40% of the test dataset, which is visible during the competition and is part of the public leaderboard. "Private" refers to the remaining 60% of the test dataset, which is only visible after the competition has ended and is part of the private leaderboard.

Image	Text	Public	Private
Eff-B3	TFIDF	0.723	0.715
ResNext50	TFIDF	0.726	0.716
Eff-B4	TFIDF	0.728	0.718
Eff-B5	TFIDF	0.730	0.720
ECA-NFL0	TFIDF	0.731	0.718

Table 2. Single Image Model + TFIDF Results

Image	Emsemble	Public	Private
ResNext50+Eff-B3	Avg	0.729	0.719
ResNext50+Eff-B3	Concat	0.729	0.719
Eff-B5+ECA-NFL0	Concat	0.732	0.722

Table 3. Multiple Image Models + TFIDF Results

Image	Text	Fusion	Public	Private
ECA-NFNet-L0	TFIDF	Union	0.731	0.718
DT (ECA-NFNet-L0)	DT (TFIDF)	Union	0.734	0.721
DT (ECA-NFNet-L0)	TFIDF	Union	0.738	0.726
DT (ECA-NFNet-L0 + TFIDF)		-	0.745	0.734
DT (ECA-NFNet-L0 + TFIDF) + SBert		-	0.746	0.735
DT (ECA-NFNet-L0 + TFIDF + SBert)		-	0.748	0.736

Table 3. Dynamic Threshold Results
(DT(*) indicates dynamic thresholds to * , '+' indicates Union)

Image Model: In Table 2, the results of a single CV (Computer Vision) model combined with TFIDF (Term Frequency-Inverse Document Frequency) are presented as a union. In the public leaderboard, ECA-NFNet-L0 achieved the best score of 0.731, while EfficientNet-B5 achieved the best score of 0.720 in the private leaderboard.

In Table 3, the results are shown for using two CV models combined with TFIDF as a union. It can be observed that ensemble methods, both averaging and cascading, yield similar results. The cascade of EfficientNet-B5 and ECA-NFNet-L0 slightly improves the score by 0.001 compared to using a single model. The minor improvement is likely due to the use of similar training methods, including loss functions and resolution.

Dynamic Threshold: Table 4 explores the effects of dynamic thresholding. It is observed that when dynamic thresholding is applied separately to the CV models and TFIDF, and the results are then combined as a union, there is no significant improvement. However, when the dynamic threshold is directly applied to the combined predictions of both (as mentioned in Section 3.3), the score shows a substantial increase (Public Score: from 0.731 to 0.745).

The reason is that the dynamic threshold setting exits when the predicted results exceed 1. Merging within dynamic thresholds ensures a balance between precision and recall, becoming a key factor for improving scores in the competition. Furthermore, when SBert is included in the dynamic threshold (DT) setup, the public leaderboard score reaches 0.748, and the private leaderboard score is high enough to rank within the top 70.

Query Expansion: In addition to dynamic thresholding, Query Expansion (QE) has become another crucial method for score improvement. Table 5 explores the effects of αQE, where "++" indicates that two CV models are integrated through concatenation. From the table, it's clear that using αQE on image features significantly improves the score, regardless of whether the Text part is included in the dynamic thresholding or not. This improvement is consistent across both Public and Private leaderboards.

The final submission model employs a cascaded version of EfficientNet-B3, EfficientNet-B5, and ECA-NFNet-L0. Compared to the cascading of just two CV models, this setup shows a slight improvement in results.

Image	Text	Fusion	αQE	Public	Private
DT(ECA-NFNet-L0++EfficientNet-B5)	TFIDF	Union	×	0.739	0.726
DT(ECA-NFNet-L0++EfficientNet-B5)	TFIDF	Union	✓	0.745	0.732
DT ((ECA-NFNet-L0++EfficientNet-B5)+TFIDF+SBert)		-	×	0.749	0.737
DT ((ECA-NFNet-L0++EfficientNet-B5)+TFIDF+SBert)		-	✓	0.756	0.743

Table 5. αQE Results
('++' indicates features from two models are cascaded)

5 Conclusion

This paper provides a retrospective analysis of the solution for the 2021 Kaggle "Shopee-Price Match Guarantee" competition, discussing the approaches in terms of image, text, and post-processing. The main opportunity for score improvement in this competition lies in the correct application of post-processing methods. Ultimately, we achieved an F1-Score of 0.756 on the public dataset, ranking 36th out of 2,426; and an F1-Score of 0.744 on the private dataset, ranking 39th out of 2,426.

In addition to the techniques discussed in this paper, further improvements can be made by first cascading image and text features before calculating similarity. Unfortunately, one issue that remained unresolved in this competition is ensuring that all samples within the same group should have the same predicted results (i.e., if A predicts B and C, then B should also predict A and C, and C should also predict A and B). In the future, we hope to find more effective methods to address this issue, such as using graph-based relational modeling.

References

[1] New deep learning optimizer, ranger: Synergistic com- bination of radam + lookahead for the best of both, 2020. https://github.com/lessw2020/Ranger-Deep- Learning- Optimizer. 5

[2] Kaggle"shopee-price match guarantee", 2021. https: //www.kaggle.com/c/shopee- product-matching/ overview. 1

[3] A. Babenko and V. Lempitsky. Aggregating deep con- volutional features for image retrieval. arXiv preprint arXiv:1510.07493, 2015. 2, 4

[4] A. Brock, S. De, S. L. Smith, and K. Simonyan. High-performance large-scale image recognition with- out normalization. arXiv preprint arXiv:2102.06171, 2021. 2

[5] J. Deng, J. Guo, N. Xue, and S. Zafeiriou. Arcface: Additive angular margin loss for deep face recogni- tion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4690–4699, 2019. 3, 5

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transform- ers for language understanding. arXiv preprint arXiv:1810.04805, 2018. 2

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep resid- ual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016. 2

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Im- agenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25:1097–1105, 2012. 1

[9] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. Sphereface: Deep hypersphere embedding for face recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 212– 220, 2017. 3

[10] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101, 2017. 5

[11] F. Radenović, G. Tolias, and O. Chum. Fine-tuning cnn image retrieval with no human annotation. IEEE transactions on pattern analysis and machine intelli- gence, 41(7):1655–1668, 2018. 2, 4, 5

[12] J. Ramos et al. Using tf-idf to determine word rele- vance in document queries. Citeseer, 2003. 2

[13] N. Reimers and I. Gurevych. Sentence-bert: Sen- tence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084, 2019. 2

[14] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recog- nition challenge. International journal of computer vi- sion, 115(3):211–252, 2015. 1

[15] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and cluster- ing. In Proceedings of the IEEE conference on com- puter vision and pattern recognition, pages 815–823, 2015. 2

[16] K. Simonyan and A. Zisserman. Very deep convo- lutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014. 2

[17] Y. Sun. Deep learning face representation by joint identification-verification. The Chinese University of Hong Kong (Hong Kong), 2015. 2

[18] Y. Sun, C. Cheng, Y. Zhang, C. Zhang, L. Zheng, Z. Wang, and Y. Wei. Circle loss: A unified perspec- tive of pair similarity optimization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6398–6407, 2020. 3

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabi- novich. Going deeper with convolutions. In Proceed- ings of the IEEE conference on computer vision and pattern recognition, pages 1–9, 2015. 2

[20] M. Tan and Q. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In Interna- tional Conference on Machine Learning, pages 6105– 6114. PMLR, 2019. 2

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In Advances in neural in- formation processing systems, pages 5998–6008, 2017. 2

[22] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu. Cosface: Large margin cosine loss for deep face recognition. In Proceedings of the IEEE conference on computer vision and pattern recogni- tion, pages 5265–5274, 2018. 3

[23] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discrimi- native feature learning approach for deep face recog- nition. In European conference on computer vision, pages 499–515. Springer, 2016. 2