# FC-Aligner: A Lightweight Regressor Model for Embedding Space Conversion

**André Luiz Vieira-e-Silva** **René Ferrari** **Álvaro Nolibos** **Gustavo Zanoni Felipe**

AI Biometrics, Caf*
R. Tiradentes, 1077, Venâncio Aires, Brazil
{andre.silva, rene.ferrari, alvaro.nolibos, gustavo.felipe}@caf.io

## Abstract

In diverse applications like image clustering, facial recognition and text embeddings, similarity search is critical. Deep models utilize feature embeddings for efficient representation, learning shared similarities during training. However, developing new models raises compatibility issues, necessitating the re-extraction of the entire embedding database (backfilling). Very large datasets become a bigger problem, considering the necessary time and computational power. While solutions like backward compatibility models and harmonic embeddings exists, they may be impractical without the original input data. We present a simple yet efficient model called FC-Aligner, which converts embeddings from a previous embedding space to a new one through a regression-inspired approach. We use production data from a real-world face recognition system, with a total of 3.39M samples. Results show an acceptable increase in FAR from $0.0032\%$ to $0.0048\%$, while keeping a similar FRR ($\sim 5\%$). Using FC-Aligner in 20M embeddings is $11\times$ faster and $2.5\times$ cheaper than backfilling.

## 1 Introduction

The use of feature embeddings is prevalent in large language models and computer vision, providing vectorized representations of text and images. These embeddings enable similarity assessments, such as matching face images to determine if they belong to the same person (10).

Various deep learning models have been developed to generate embeddings from face images (10; 7; 14). In our case, the images to be encoded by the embedding generator model consist of cropped and aligned human faces. The process of encoding all the images in a database is known as backfilling. A gallery set consists of all encoded images, and the task of matching face embeddings (similarity distance under a defined threshold) is recognized as face recognition.

As model architectures, datasets, and loss functions evolve, new embedding generators may outperform previous ones, necessitating a costly backfilling of existing datasets to ensure compatibility in the latent space. This challenge is well-known in both academia and industry, and while various solutions have emerged (8; 13; 5; 6), it is still an open challenge.

To achieve embedding compatibility, strategies include direct compatibility training during model training (10), or developing translation functions for embeddings between spaces (13; 5). Compatibility can be backward or forward, where the latter enables translating old embeddings to a new embedding space.

*https://www.caf.io/

Latinx in AI @ NeurIPS-24 (LXAI 2024).

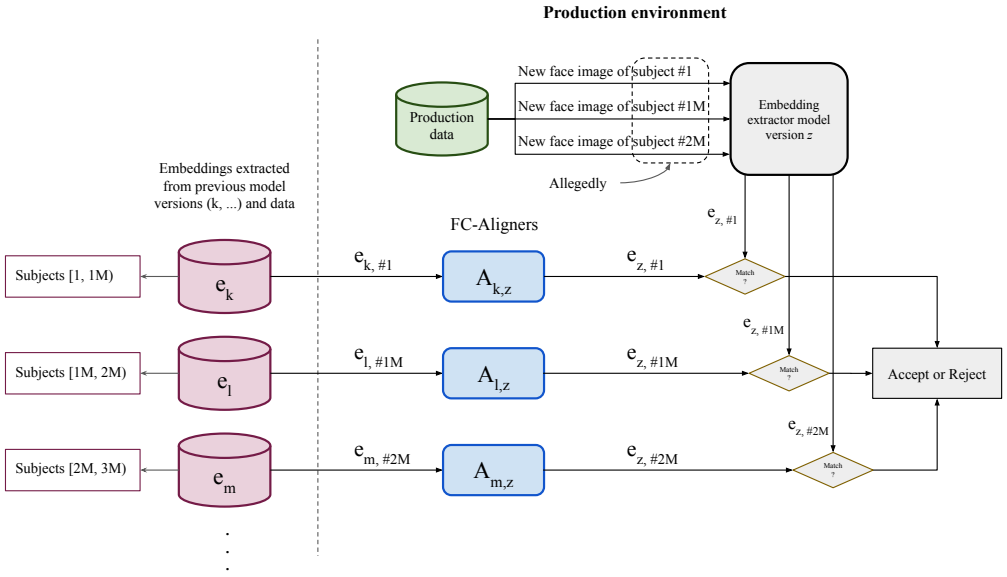



Figure 1: An overview of our proposed framework. To avoid backfilling, we train multiple lightweight and fast forward-compatibility models to align embeddings extracted from previous models with embeddings from the latest version of our embedding extractor model. Considering the employed Face Recognition application, embeddings extracted from different face images using different embedding models are now comparable with little degradation and a significant cost reduction.

To mitigate the costs associated with backfilling, we propose a lightweight neural network that translates old embeddings into new latent spaces while preserving face-matching metrics. Our proposed framework is summarized in Figure 1.

This work hypothesizes that a model can effectively convert embeddings from an older version of a Face Recognition (FR) model to the embedding space of a newer version. The old and new versions of the FR model differ in the used training dataset and hyperparameters, but not in network architecture. The conversion model should be lightweight and fast, making it a cost-effective alternative to re-extracting embeddings (backfilling) with the new FR model. It should also require only a small sample size for training compared to the entire embedding database.

We propose FC-Aligner, a lightweight model for converting face embeddings between different versions of an FR model. Using FC-Aligner, we could have avoided a costly backfilling process for a dataset of over 20 million faces, which increased cloud expenses by 287.5% for storage and 67.32% for computing, taking 5 straight days to complete. Instead, FC-Aligner required only an 8-hour training session on a commercial GPU, converting all embeddings in 3 hours on a CPU-only laptop. The False Acceptance Rate increased by just 0.0016 percentage points, while maintaining a similar False Rejection Rate of around 5%.

## 2 Related Works

**Backfilling.** Each newly trained model creates a new embeddings space (or latent space) that can not be compared to the embeddings generated by an old model, discarding all the processing time invested to generate the old embeddings. When this happens, all the old subjects must have their embeddings regenerated by the new model. This process, called backfilling, it is very expensive in terms of processing time and financial costs. That is why a lot of works come with different approaches to improve this method, like (6; 12; 11) that came up with a faster approach based on making only a partial backfilling.

**Model Compatibility.** Since backfilling is expensive, some techniques emerged with the goal of making the embeddings compatible across models. There are many ways to achieve this, the following being the most common currently.

**Direct Compatibility.** One way to guarantee the compatibility between embeddings generated by two models is using the embeddings generated by the old model during the new model training (8; 15; 10), using the difference between the new and old embeddings in the loss function. This way, the new model is forced to focus not only on making the embeddings of the same subject to be near each other but also to be near the embeddings generated by the old model. This approach is used by (8) in its compatibility framework to deal with Compatible Training cases. We performed some experiments inspired by the harmonics embeddings (10) approach. However, in the context of this research, we obtained a decrease in the accuracy of the new model.

**Embedding Compatibility.** Another way to achieve compatibility is to use some transformation approach for converting the embeddings from one embedding space into another. This can be used in both ways, such from the old embedding space to the new one, known as Forward Compatibility (9; 3), and the other way around, known as Backward Compatibility (13; 5). (8) also uses both approaches in its framework.

**Backward Compatibility.** (13) proposed a framework called Backward-Compatible Training (BCT), which can train models that are compatible with previously computed embeddings regardless of whether they have different dimensions, learned through different network architectures and losses.

**Forward Compatibility.** (9) presented a new paradigm, known as Forward Compatible Training (FCT), where, during the training, the current model prepared the embeddings to be updated in the future when a new model will be trained. This was made by preserving some features that may not be useful for the current embedding representation, but will facilitate its transformation to a new embedding space. In addition, (3) was successful in transforming voice profiles between embedding spaces, making it possible to use multiple models for voice identification at the same time.

With all of that in mind, the main challenge is to develop a lightweight model that is faster and cheaper than backfilling, while minimizing accuracy loss and the gap between embedding spaces. Given our continuous influx of new anonymized data, a forward compatibility strategy is more appropriate. Therefore, we propose the FC-Aligner model to efficiently adapt to future models.

# 3 Methodology

## 3.1 Problem Setting

We ground our application to face recognition despite our formulation being general and applicable to other domains, such as other computer vision tasks and natural language processing. First, we define a model that maps face images into face embeddings $M_k : F \rightarrow E_k$, where $k \in \mathbb{N}$ is the model's version. In other words, a model $M_k$ receives a face image $f_i \in F$, where $i$ is an image unique identifier, and returns $e_{k,i} \in E_k$, where $e_{k,i}$ means "the embedding of face image $i$ extracted with model version $k$" and $E_k$ is the embedding space yielded by model $M_k$. Different model versions, such as $M_j$ or $M_l$, where $j < k$ and $l > k$, mean older and newer model versions, respectively. Two embeddings extracted with the same model $M_k$ from different images, e.g., $f_0$ and $f_1$, are considered embeddings from the same subject (person) if their Euclidean distance is lower than some threshold.

During training, for every iteration, there is an anchor embedding, a positive embedding, which is an embedding extracted from a different image of the same subject as the anchor, and a negative embedding, which is an embedding extracted from a different subject. Hence, a new model $M_{k+1}$ is learned by using a batch hard triplet loss (4), which approximates the anchor embedding to the "hardest positive", which is the positive embedding farthest from the anchor inside that batch, and distances the anchor to the "hardest negative", which is the negative embedding closest to the anchor inside that batch. In our study, new embedding extraction models are proposed by changing the dataset (size and samples), the training parameters and hyperparameters, such as loss function, optimizer, learning rate, and batch size, although the network architecture remains constant, a residual convolutional neural network.

Embeddings extracted from different model versions, for instance, $e_{k,0}$ and $e_{l,1}$ are incompatible and may not be compared. Hence, if a new model $M_{k+1}$ is developed, all embeddings have to be re-extracted from the entire face image dataset, even the ones that already have been extracted before.

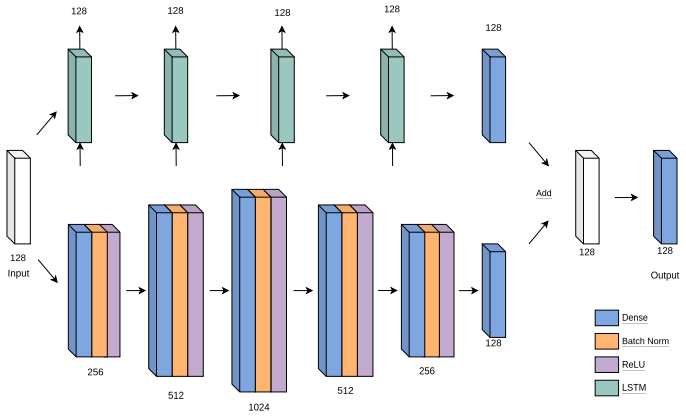



Figure 2: FC-Aligner multi-branch architecture.

### 3.2 Forward Compatibility Alignment

FC-Aligner is a lightweight regressor model $A$ that maps embeddings from one embedding space $E_k$ to a newer embedding space $E_l$ ($l > k$) as: $A_{k,l} : E_k \rightarrow E_l$. In other words, $e_{k,i} \in E_k$ can be mapped to $e_{l,i} \in E_l$ and may be compared to other embeddings from $E_l$. Therefore, an image $f_1$, that has its embeddings extracted from a newer model $M_l$ can be compared to an embedding extracted *a priori* from another image $f_0$, with a model $M_k$ using a distance function $dist$ and a threshold $th$ by:

$$dist(A_{k,l}(e_{k,0}), M_l(f_1)) < th, \tag{1}$$

where $l > k$. Two embeddings extracted *a priori* can also be aligned to a newer, third embedding space $E_m$ to be compared, as in:

$$dist(A_{k,m}(e_{k,0}), A_{l,m}(e_{l,1})) < th. \tag{2}$$

We propose lightweight architectures to perform the forward compatibility alignment, each prioritizing a different aspect: FC-Aligner$_{dense}$ and FC-Aligner$_{mb}$; in the latter, $mb$ refers to the fact the architecture has multiple branches. Figure 2 shows a graphical representation of the FC-Aligner$_{mb}$ architecture. The FC-Aligner$_{dense}$ architecture is basically the second branch of FC-Aligner$_{mb}$.

#### 3.2.1 Training setup

First, we sample from embedding databases whose embedding spaces we want to generate an aligner, e.g., $k$ and $l$, where $k < l$. We now produce a training dataset with tuples of old and new embeddings that correspond to the embeddings extracted with the old and new models from the same image, i.e., $(e_{k,0}, e_{l,0}), (e_{k,1}, e_{l,1}), ..., (e_{k,i}, e_{l,i})$, where $i$ is the dataset size. We learn $A_{k,l}$ by minimizing the training loss, which is the Euclidean distance between an old and a new embedding of an input image.

#### 3.2.2 Testing setup

For testing, we sample from the same databases, but to produce the testing dataset, anchor and positive embedding pairs are formed, i.e., $(e_{k,anc_0}, e_{k,pos_0}), ..., (e_{k,anc_i}, e_{k,pos_i}), (e_{l,anc_0}, e_{l,pos_0}), ..., (e_{l,anc_i}, e_{l,pos_i})$, where $k$ and $l$ are embedding extraction model versions, and $i$ is the dataset size. Now, we have pairs of embeddings extracted from different face images of the same subject using two embedding models.

The evaluation is done in an $N : N$ fashion, following four scenarios. Considering embedding$_{new}$ as an embedding generated by a target model and embedding$_{converted}$ as an embedding generated from a source model and converted to the target space by FC-Aligner, we define those four scenarios as a comparison of: 1. anchor$_{new}$ vs. positive$_{new}$, 2. anchor$_{converted}$ vs. positive$_{converted}$, 3. anchor$_{converted}$ vs. positive$_{new}$, and 4. anchor$_{new}$ vs. positive$_{converted}$. It is worth mentioning that

these scenarios aim to simulate a real-world application of FC-Aligner, where converted embeddings are compared to newly generated ones.

Each embedding has exactly three matches and $\sum_{n=1}^{ds} n$ non-matches, where $ds$ is the total dataset size or the total amount of embeddings in the test set. For example, following the terminology described above, $A_{k,l}(e_{k,anc_0})$ has exactly three matches: $A_{k,l}(e_{k,pos_0})$, $e_{l,anc_0}$, and $e_{l,pos_0}$.

#### 3.2.3 Inference setup

At inference time, the model receives one or a batch of embeddings from a source embedding space and aligns it or them to a destination embedding space. In a production environment, multiple FC-Aligners could co-exist, aligning embeddings from different source spaces to the space of the current embedding model in production, as shown in Figure 1.

## 4 Experimental Setup

### 4.1 Metrics, Parameters and Infrastructure

Following the evaluation scenarios described in Section 3.2, we employ an $N : N$ evaluation approach that compares every anchor embedding to the other embeddings in the test set that do not belong to the same identity (person). Such comparison is based on calculating the euclidean distance between embeddings and comparing the calculated distance to a pre-established threshold. A match is characterized by when such distance is smaller than the threshold. Otherwise, we call it a non-match.

By calculating the False Negatives ($fn$), False Positives ($fp$), True Negatives ($tn$), and True Positives ($tp$) values, we are able to evaluate our overall performance considering the False Rejection Rate (FRR): $FRR = \frac{fn}{fn+tp}$ and False Acceptance Rate (FAR): $FAR = \frac{fp}{fp+tn}$.

In the face recognition scenario, FRR measures the probability of a false non-match, i.e., the rejection of two faces from the same identity. As FAR estimates the rate of acceptance (match) of two different identities. When dealing with production metrics, such value would also represent how susceptible the model is to facial identity fraud.

Models and training/testing pipelines were developed in TensorFlow 2 (1), with 240 maximum epochs with a 10-epoch early stopping. We used the Adam optimizer, a $10^{-3}$ learning rate with 2-epoch patience to be reduced a factor of 0.1. Batch sizes of 128 or 512 elements were evaluated. Training sessions were taken on the AWS EC2 environment using a $g4dn.xlarge$ instance which has 4 vCPUs, 16GB of RAM, and 1 GPU NVIDIA T4 with 16GB of VRAM.

FaceNet embeddings are typically trained with the Triplet Loss function, suggesting their correlations exist in Euclidean space. Therefore, we used Euclidean Distance as a loss function, expecting the model to learn to regress embeddings from the source to the target space, resulting in higher loss values when embeddings diverge from their target points.

### 4.2 Datasets

The used dataset is composed of approximately 3.36 million embedding pairs for training/validation and 14,848 pairs for testing the developed models. All embeddings were generated from a randomly selected pair samples of face images from a private set of samples originating from an internal identity solution. Due to the high sensitivity of this data, no real sample will be presented in this paper. However, they can be described as arrays of 128 floating-point numbers. During training, 17.5% of the full dataset was split for validating the model and generating online metrics.

To obtain the embedding images, two FaceNet (10) models were adapted following the training schema described in the original paper. Both of them were developed with the goal of simulating two stages of a deep learning model life-cycle. The first one was trained with a smaller dataset that consisted of $\sim$200K face pairs from distinct identities. The second one was trained with a larger dataset composed of $\sim$1M pairs of faces. Such an approach was taken, considering the earliest version of a production model in highly growing companies is usually built with limited resource. In the later iterations of such a model, the increase in data samples implies an overall quality improvement.

Table 1: Results table of multiple FC-Aligner models. Three comparable FC-Aligner models were trained under varied parameters/hyperparameters to convert embeddings from model version 11 to 46.4. For a 'balanced FRR-FAR' objective, the FC-Aligner Dense architecture is more fit, whereas, for a 'lowest FAR' goal, the Multi-branch architecture is more suitable. For the other tuple of model alignments, the Multi-branch approach shows the best results for both objectives.

| | Old\|New | Architecture | Train Set Subjects | Batch Size | Scenario | Same Subj Avg Distance | Different Subj Avg Distance | FRR | FAR |
|---|---|---|---|---|---|---|---|---|---|
| Best FRR-FAR relation | 11\|46.4 | Dense | 1.68M | 512 | 1 | 0.902679 | 2.183250 | 4.977101% | 1/30775 |
| | 11\|46.4 | Dense | 1.68M | 512 | 2 | 0.857281 | 1.954131 | 2.929688% | 1/11082 |
| | 11\|46.4 | Dense | 1.68M | 512 | 3 | 0.957023 | 2.071042 | **5.293642%** | **1/20779** |
| | 11\|46.4 | Dense | 1.68M | 512 | 4 | 0.956371 | 2.071352 | 5.428341% | 1/20561 |
| Lowest FAR | 11\|46.4 | Multi-branch | 1.68M | 512 | 1 | 0.902679 | 2.183251 | 4.977101% | 1/30775 |
| | 11\|46.4 | Multi-branch | 1.68M | 512 | 2 | 0.864956 | 1.975253 | 3.077856% | 1/12110 |
| | 11\|46.4 | Multi-branch | 1.68M | 512 | 3 | 0.961525 | 2.081599 | 5.697737% | 1/21765 |
| | 11\|46.4 | Multi-branch | 1.68M | 512 | 4 | 0.961273 | 2.081967 | 5.832435% | 1/21572 |
| | 11\|46.4 | Multi-branch | 321k | 128 | 1 | 0.902679 | 2.183251 | 4.977101% | 1/30775 |
| | 11\|46.4 | Multi-branch | 321k | 128 | 2 | 0.909361 | 2.046864 | 5.650593% | 1/21091 |
| | 11\|46.4 | Multi-branch | 321k | 128 | 3 | 0.974820 | 2.115910 | 7.132274% | **1/27607** |
| | 11\|46.4 | Multi-branch | 321k | 128 | 4 | 0.973671 | 2.115586 | 7.293912% | 1/27263 |
| | 3\|11 | Multi-branch | 60k | 128 | 1 | 0.555701 | 1.264120 | 4.533921% | 1/23523 |
| | 3\|11 | Multi-branch | 60k | 128 | 2 | 0.497169 | 1.155949 | 2.196848% | 1/1930 |
| | 3\|11 | Multi-branch | 60k | 128 | 3 | 0.583338 | 1.210613 | 4.373665% | 1/5263 |
| | 3\|11 | Multi-branch | 60k | 128 | 4 | 0.582165 | 1.210958 | 4.033120% | 1/5423 |
| | 11\|16 | Multi-branch | 60k | 128 | 1 | 0.676062 | 1.571831 | 4.983836% | 1/24523 |
| | 11\|16 | Multi-branch | 60k | 128 | 2 | 0.657643 | 1.533731 | 3.683998% | 1/17246 |
| | 11\|16 | Multi-branch | 60k | 128 | 3 | 0.683286 | 1.552855 | 4.404634% | 1/20716 |
| | 11\|16 | Multi-branch | 60k | 128 | 4 | 0.683358 | 1.552975 | 4.552802% | 1/20877 |
| | 11\|27 | Multi-branch | 60k | 128 | 1 | 0.713236 | 1.643047 | 4.990571% | 1/28401 |
| | 11\|27 | Multi-branch | 60k | 128 | 2 | 0.683606 | 1.578432 | 3.158675% | 1/14921 |
| | 11\|27 | Multi-branch | 60k | 128 | 3 | 0.730033 | 1.610853 | 4.424838% | 1/21488 |
| | 11\|27 | Multi-branch | 60k | 128 | 4 | 0.730411 | 1.611267 | 4.431573% | 1/21484 |

Finally, the above-mentioned 3.36 million embeddings are generated from both models. The target model, i.e., trained with a larger dataset, usually outperforms the source model in accuracy. In our experimentation scenario, the source model reached a FAR of 1/100k, while the target model presented $1/2$M for the same metric. Both of these values were obtained considering a fixed FRR of 5%. Keeping that under consideration, the high increase in performance makes the target model suitable for deployment. Supporting the opportunity of making the transition and avoiding backfilling.

## 5 Results and Discussion

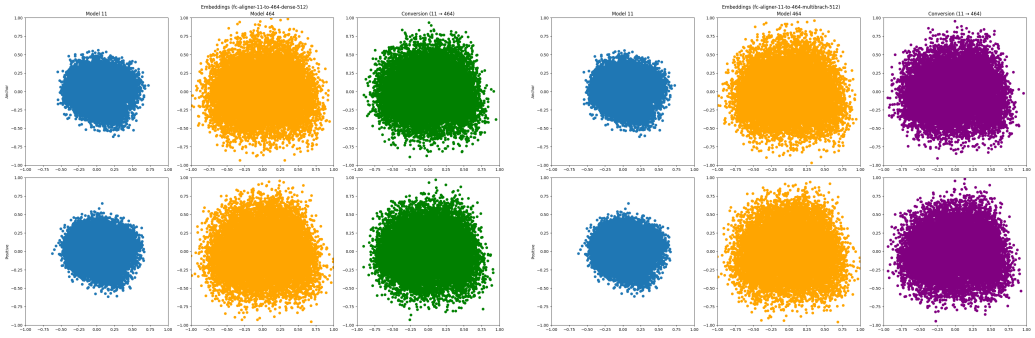
### 5.1 Quantitative analysis

This study aims to evaluate the performance of the FC-Aligner architectures under different conditions. To do this, we mainly compare false rejection rates and false acceptance rates. We also show how the different conditions impact the average distance of embeddings belonging to the same subject and also to different subjects. The results are presented in Table 1 and are analyzed below.

Our findings indicate that the best FRR-FAR relation for an aligner between versions 11 and 46.4 is achieved using the FC-Aligner Dense architecture and a training set size of 1.68M subjects (3.36M embeddings). From our experiments, the increased batch size of 512 was essential to reach an acceptable FRR-FAR relation, as a good FRR-FAR can also be observed in the setting directly below using the multi-branch approach.

However, when we focused on the False Acceptance Rate (FAR) alone, we noticed that the FC-Aligner Multi-branch architecture provided the best results as evidenced in all trained aligners: $A_{3,11}$, $A_{11,16}$, $A_{11,27}$, and $A_{11,46.4}$. This highlights the versatility of the FC-Aligner architectures, as different versions can be optimized for different performance metrics.

Experimenting using different batch sizes produced more balanced results, as evidenced by the achieved FRR-FAR relation. When aiming for a lower FAR only, a smaller batch size was more suitable. The dataset size also had a positive impact when aiming for a better FRR-FAR relation and



(a) Old vs. New vs. Converted (Dense). (b) Old vs. New vs. Converted (Multi-branch).

Figure 3: Individual scatter embedding plots enabled by two-component PCA. It is notable the scatter pattern changes from Model 11 (blue) to the aligned ones (green and purple), better matching the pattern shown by the embeddings extracted from Model 46.4.

for lower individual metrics in general, as evidenced by the results achieved using small datasets of 60k subjects, which fell short of the results achieved using large datasets.

We also evaluated the models in different scenarios, as shown in the Testing setup paragraph of Section 3.2. It is important to mention that Scenario 1 compares embeddings that were extracted directly using the target model, so it is used as a baseline reference of the maximum achievable performance. It is important to note that we use 'target model' and 'new model' as synonyms.

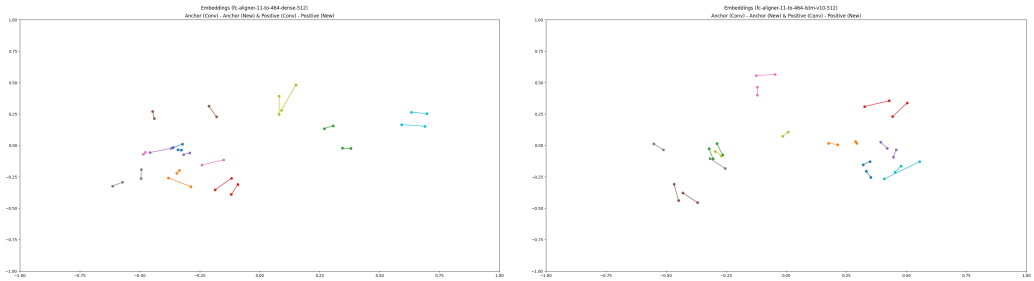
Our results show that Scenarios 3 and 4 consistently perform better across different versions and architectures in terms of FRR-FAR relation and FAR alone. This suggests that using at least of embedding extracted with the target model can provide superior results. Following our proposed framework logic shown in Figure 1, Scenario 3 is the one closest to reality if we consider that the aligned embedding from the database is the anchor and the newly extracted embedding from the production data is the positive.

### 5.2 Qualitative analysis

This analysis aims to provide graphical evidence that the FC-Aligner conversions perform an embedding space conversion and not only reach promising numerical results. For this, we apply the Principal Component Analysis (PCA) technique to reduce the embedding dimensionality from 128 to 2 so that it can be plotted in 2D space. Figure 3 shows the test set embedding scatter plots, in which the first two columns were obtained by direct extraction from embedding models and the last column by converting the source embeddings using the two different FC-Aligner architectures.

In Figure 3, it is clear the changes in the scatter pattern from Model 11, in blue, to the two converted ones, in green and purple. It is noticeable that the new scatter pattern obtained by using FC-Aligner is closer to the one presented by the embeddings directly extracted from target Model 46.4.

Finally, Figure 4 shows some plotted embedding samples extracted with the target model and converted samples from the source model. Each figure shows converted samples from the two proposed aligner architectures. In it, there are ten sets of four points, in which each set of four points have the same color, representing embeddings from images of the same subject. In those sets, there are two pairs connected by a straight line, meaning they are embeddings of the same image but extracted differently: one by the target model and the other by converting the source model extraction with an FC-Aligner model. It is possible to note that, in general, points of the same color are close to each other and connected points are even closer, highlighting once again the converted embeddings are not only close to each other but are also close to different embeddings from different input face images of the same subject.



(a) Target and converted embeddings samples (Dense). (b) Target and converted embeddings samples (M-B).

Figure 4: PCA plots of the target model and converted embedding samples for both architectures. Points of the same color (in ten sets of four) represent embeddings from the same subject, while connected points indicate embeddings extracted from the same image.

### 5.3 Inference speed and memory usage

In the inference speed experiment, we utilized embedding data from Model 11, processing batches of 8192 on a CPU-only laptop (i7-1165G7 @ 2.80GHz, 16 GB RAM). The results showed that FC-Aligner's Dense architecture converted 317,846 subjects in 176.8 seconds, while the Multi-branch architecture took 205 seconds. The Dense model produced a 5.4 MB Keras `.h5` file, compared to 7.6 MB for the Multi-branch model.

Both models were converted to ONNX (2) without size changes. Using the ONNX model, the Dense approach converted embeddings in 52.2 seconds (approximately 3.4 times faster), with an average Euclidean distance of $4.67 \times 10^{-7}$. The Multi-branch approach took 49 seconds (about 4.2 times faster), with an average distance of $4.93 \times 10^{-7}$. ONNX optimization allows for embedding conversion three times faster with minimal precision loss.

For 20 million embeddings, the Dense approach took 3 hours with Keras, while the ONNX model reduced this to 55 minutes. Compared to the 8-hour GPU training time and 5 days for full backfilling, FC-Aligner is at least 10.8 times faster, with the ONNX model achieving over 13 times the speedup.

### 5.4 Limitations

Although FC-Aligner saves a significant amount of time and computational costs, it is not sustainable to maintain and keep generating more FC-Aligner models if new embedding extraction models are being constantly developed and deployed into production. Taking into consideration all constraints raised in this work, it would be difficult to propose a universal aligner, although ideal.

## 6 Conclusion

In this work, we proposed a method to employ forward compatibility in embedding-generation models, more specifically for face recognition, in order to avoid large database backfilling. As discussed throughout the paper, the process of backfilling is greatly expensive in many aspects. In many scenarios, it requires the usage of large computational resources in order to fulfil the goal of updating the data entries from one version of an embedding model to another. Our method is based on a lightweight model, namely FC-Aligner, capable of regressing an embedding from a source latent space to a target one. The found results are promising; by applying our multi-branch version of the model, we were able to convert a large face database $11\times$ faster than a regular backfilling, saving at least $2.5\times$ the cost of performing such action. When it comes to performance metrics, we were able to maintain a similar FRR while presenting a tolerable metric decay on our final FAR value.

The next works will aim to further expand the concept of the FC-Aligner for different applications. We look forward to testing the here developed model for text embeddings (LLM-based models, e.g., text similarity search) and other application tasks. Also, we consider validating other architectures as a way to improve our method, with the goal of reducing the conversion degradation.

## References


[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), `https://www.tensorflow.org/`, software available from tensorflow.org

[2] developers, O.R.: Onnx runtime. `https://onnxruntime.ai/` (2021), version: x.y.z

[3] Gao, C., Desplanques, B., Ju, C.J.T., Chadha, A., Stolcke, A.: Post-training embedding alignment for decoupling enrollment and runtime speaker recognition models. arXiv preprint arXiv:2401.12440 (2024)

[4] Hermans, A., Beyer, L., Leibe, B.: In defense of the triplet loss for person re-identification. arXiv preprint arXiv:1703.07737 (2017)

[5] Hu, W., Bansal, R., Cao, K., Rao, N., Subbian, K., Leskovec, J.: Learning backward compatible embeddings. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 3018–3028 (2022)

[6] Jaeckle, F., Faghri, F., Farhadi, A., Tuzel, O., Pouransari, H.: Fastfill: Efficient compatible model update. arXiv preprint arXiv:2303.04766 (2023)

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

[8] Meng, Q., Zhang, C., Xu, X., Zhou, F.: Learning compatible embeddings. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9939–9948 (2021)

[9] Ramanujan, V., Vasu, P.K.A., Farhadi, A., Tuzel, O., Pouransari, H.: Forward compatible training for large-scale embedding retrieval systems. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 19386–19395 (2022)

[10] Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 815–823 (2015)

[11] Seo, S., Uzunbas, M., Han, B., Cao, X., Zhang, J., Tian, T., Lim, S.N.: Metric compatible training for online backfilling in large-scale retrieval. In: ICML Workshop on Localized Learning (LLW) (2023)

[12] Seo, S., Uzunbas, M.G., Han, B., Cao, S., Zhang, J., Tian, T., Lim, S.N.: Online backfilling with no regret for large-scale image retrieval. arXiv preprint arXiv:2301.03767 (2023)

[13] Shen, Y., Xiong, Y., Xia, W., Soatto, S.: Towards backward-compatible representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6368–6377 (2020)

[14] Shi, Y., Jain, A.K.: Probabilistic face embeddings. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6902–6911 (2019)

[15] Zhang, Y., Lu, H.: Deep cross-modal projection learning for image-text matching. In: Proceedings of the European conference on computer vision (ECCV). pp. 686–701 (2018)