Behavioral Stylometry in Chess: An Embedding Approach to Detecting Individual Decision-Making Style

Anonymous Author(s)
Affiliation
Address
email

Abstract

The advent of machine learning models that surpass human decision-making ability in complex domains has initiated a movement towards building AI systems that can interact with humans. Many building blocks are essential to this activity, and one of the central ones is the algorithmic characterization of aspects of human behavior. While much of the existing work focuses on characterizing human behavior in an aggregate or generic sense, an important longer-range goal is to specialize these characterizations to individual people.

Specializing behavioral models to individual people requires that we be able to characterize what differentiates specific people. As a means of formalizing this process, we study the problem of *behavioral stylometry*, in which the task is to identify a decision-maker from their decisions alone. We present a transformer-based approach to behavioral stylometry in the context of chess, in which a game's players are identified from their moves. Our method operates in a few-shot classification framework, and can correctly identify a player from among thousands of candidate players given only 100 labeled games with up to 98% accuracy. Even when trained on amateur play, it generalises to out of distribution samples of Grandmaster players, despite dramatic differences in the style of play between amateur and world-class players. Finally, the resulting embeddings of players and games reveal structural properties of human style in chess.

1 Introduction

The advent of machine learning models that surpass human decision-making ability in complex domains has initiated a movement towards building human-compatible AI that could interact and work with people. Many building blocks will be essential in order to achieve this far-reaching research goal, with a central one being the algorithmic characterization of aspects of human behavior. While much of the existing work focuses on characterizing human behavior in an aggregate or generic sense, an important longer-range goal is to specialize these characterizations to individual people.

Developing algorithms that can detect signatures of individual behavior, and the idiosyncrasies of individual decision-making style, would be an important advance for many reasons. First, being able to recognize an individual from characteristics of their behavior would help enable an AI system to understand how to coordinate with that individual. In general, decision-making styles can either be complementary or clashing, and a predictive characterisation of an individual's decisions would be useful for an AI system trying to adapt to a particular person's style. Second, such fine-grained individual-level characterisations would help isolate the distinguishing features of individual styles. In a domain where ML systems are able to match or surpass human performance, it becomes natural

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

to try extending these systems to be able to pinpoint types of decisions that a person is particularly good at, and types of decisions they could potentially improve. Finally, applying these algorithms to many different individuals would shed light on the structure of decision-making in the domain itself. For example, certain instances could prove particularly discriminative in detecting individual style.

In this paper, we formalise this process by tackling *behavioral stylometry*, in which the task is to identify a decision-maker from their decisions alone. We approach this task in the domain of chess, which is an ideal model system for developing human-compatible algorithms for many reasons. AI systems surpassed all human-level ability in chess at least 15 years ago. Despite this, millions of people at all skill levels are still active in the domain. Through their activity, people generate detailed traces of their decisions, which are all recorded and publicly available for processing. In chess, behavioral stylometry reduces to predicting the identity of a player from their moves in a particular set of games.

We present a transformer-based approach to behavioral stylometry in chess. We show how to adapt recent work in speech verification to the problem of detecting individual decision-making style. Working in the domain of chess requires that we address the challenge of extending these methods to the complex types of data found in the game-playing structure of chess. Applying our model in a few-shot classification setup, where only a small number of games are provided for the model to learn a representation of a player's decisions, our model can accurately predict who they are from another small sample of their games out of thousands of possible players. Compared to a previous approach, our method is far more accurate, scalable, and data-efficient. Furthermore, our resulting embeddings of players and their games reveal the structure of human style in chess, with players who are stylistically similar clustering together in the space.

2 Related Work

Speaker verification. Our work adapts recent work in speaker verification (Variani et al., 2014; Zhang et al., 2016), where given a speaker's known utterances, the task is to verify whether an unseen utterance belongs to that speaker. Recent approaches construct an embedding space, where utterances are mapped in the space and different speakers are distinguishable (Li et al., 2017; Snyder et al., 2016). These embedding approaches can also be used for downstream tasks such as voice cloning and multi-speaker speech synthesis (Gibiansky et al., 2017; Ping et al., 2018; Jia et al., 2018).

A key decision for our task is the choice of loss function, of which two families are most relevant (Chung et al., 2020). Classification-based objectives are designed to classify a speaker out of all possible speakers. Cross-entropy loss is a simple approach with many variations including angular softmax loss (Liu et al., 2017), cosine loss (Wang et al., 2018), and additive angular margin loss (Deng et al., 2019). Contrast or metric-learning based objectives instead directly optimize the contrastiveness for samples in different classes and similarity for those within the same class. Contrastive loss (Chopra et al., 2005), triplet loss (Schroff et al., 2015) and generalized end-to-end loss (GE2E) (Wan et al., 2018) are used in face verification, face clustering and speaker verification respectively. In this work, we adapt GE2E to the problem of behavioral stylometry.

Stylometry. Both speaker verification and handwriting recognition are stylometry tasks that detect and analyze authorial style. Stylometry is has historically been associated with attribution studies (Tweedie et al., 1996) and has been applied to de-anonymizing programmers from their code (Caliskan-Islam et al., 2015), identifying age and gender from blog posts (Goswami et al., 2009) and distinguishing machine-generated text from human-generated text (Bakhtin et al., 2019). It is often framed as a task to determine the similarity or distance between two samples or between features extracted from the samples (Neal et al., 2017). Handwriting recognition also has used neural methods for decades to identify individuals (Bromley et al., 1993) and development is still being made in the field (Hafemann et al., 2016, 2017). We perform behavioral stylometry, in which the task is to identify decision-makers from their decisions.

Chess. Chess has a long history in academic research as both a test bed for artificial intelligence (Silver et al., 2018) and for understanding human behaviour (Charness, 1992; Anderson et al., 2017; Biswas and Regan, 2015). Recently these two goals have begun to merge, for example in the creation of chess engines that behave in a more human-like manner (McIlroy-Young et al., 2020a; Rosemarin

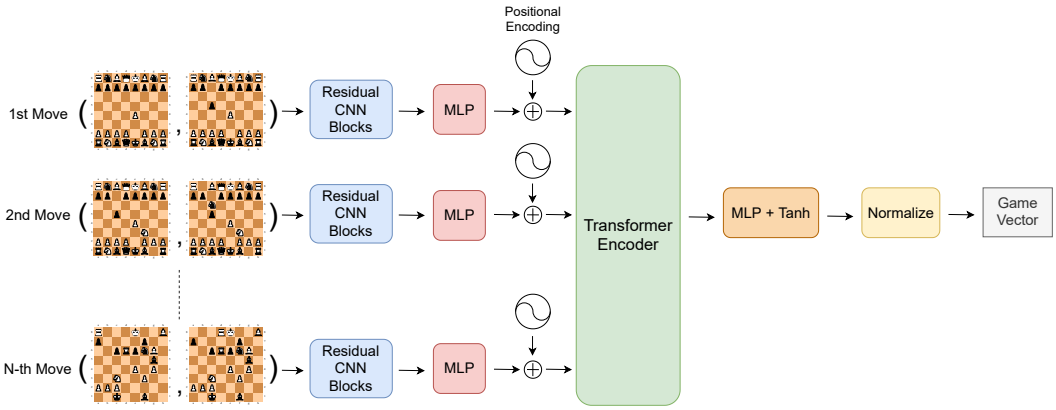


Figure 1: Overview of our model's inference procedure. It first takes in a sequence of chess moves and extracts move-level features with a series of residual blocks, then produces a game vector with a Transformer encoder followed by projection layers and normalization.

and Rosenfeld, 2019). McIlroy-Young et al. (2020b) presents personalized chess engines, based on AlphaZero (Silver et al., 2017), that are trained to predict moves made by specific players. As a side effect, they use these models to perform behavioral stylometry in chess. In this paper, we develop a methodology that is both more scalable and more accurate than this approach.

Few-shot learning. We approach behavioral stylometry as a few-shot learning task with tens to hundreds of samples per class, which is a well-studied problem formulation (Wang et al., 2020; Fei-Fei et al., 2006; Fink, 2005). Transfer learning is a main approach to few-shot learning, where models are first trained on a large dataset and then specialized to several smaller datasets (Zoph et al., 2016; Tajbakhsh et al., 2016; Donahue et al., 2014; Oquab et al., 2014). Embedding learning (Jia et al., 2014) is another approach where an embedding space is constructed so that similar training samples are close together and different samples are further apart. Test samples can then be mapped to this space and compared with the closest training samples and their associated classes. Several architectures have been proposed, such as matching networks (Vinyals et al., 2016; Bachman et al., 2017; Choi et al., 2018) and prototypical networks (Snell et al., 2017; Ren et al., 2018). Finally, meta-learning (Thrun, 1998; Andrychowicz et al., 2016) is another popular approach, which uses a two-level learning process to train a model across similar tasks and adapt it to solve new tasks with a small number of training samples (Finn et al., 2017; Ravi and Larochelle, 2017).

3 Methodology

In the section, we outline the design of our model for behavioral stylometry in chess. The high-level architecture of the model is composed of three main components: the first extracts features from the move level, the second aggregates information from the move level to the game level, and the third aggregates information from the game level to the player level. Given a chess position, the first step uses residual blocks to extract move-level features. Then in the second step, we aggregate move-level features into a game representation with a Transformer-based method. Finally, the third step aggregates a player's game vectors by calculating their centroid to produce a player vector that captures the target player's individual playing style.

Move-level feature extraction. For extracting features from chess boards and moves, we drew on methods used in previous work—AlphaZero and Maia—and use a series of residual blocks. The input to the model is a 34-channel 8×8 chess move representation that is transformed from each move in a game. We use two board positions to represent a move - the board before the player's move and the board after the move. Each board is encoded with 13 channels, with 12 channels representing each type of chess pieces for both sides and 1 channel for repetition. The extra 8 channels will encode castling rights, the side of the active player, rule 50 count and border information about the chess boards. The output is a 320-dimensional feature vector that characterizes each move.

Each residual block consists of convolutional layers, batch normalization, ReLU activations, and squeeze-and-excitation (SE) blocks (Hu et al., 2018). On top of the residual blocks we add global average pooling, which summarizes the output from residual blocks into a $1 \times 1 \times 320$ feature map. This is then flattened into a 320-dimensional vector. The hyperparameters of the model were chosen to match the Maia architecture, which uses 6 residual blocks with channel size of 64 and a reduction ratio for SE blocks of 8.

Aggregating information to the game level. The next component of the model constructs a game representation by encoding a sequence of move features that are extracted from the first component of the model. These move features correspond to the moves played by the target player in a particular game and serve as input to the model. The output will be a game vector that aggregates information about the move features. Although previous work uses an LSTM to perform the analogous task in speaker verification, our experimentation revealed that a Transformer (Vaswani et al., 2017) worked better for our task. Hence, our approach uses a modified Transformer architecture, the Vision Transformer (ViT) (Dosovitskiy et al., 2021), which was originally proposed to replace CNNs with a pure Transformer in vision tasks, and achieved comparable results to state-of-the-art CNN architectures with fewer computational resources required in training. As we consider chess moves as 8×8 images with 34 channels, we can adapt ViT to fit in our setting.

We firstly modified the input to ViT, which will take in a sequence of move features instead of fixed-size patches split from an image. Both BERT (Devlin et al., 2019) and ViT append a classification token to the input sequence, and the final hidden state of this token is considered as a representation of the entire sequence. However, in our task we have found it to work better if we directly average the output vectors of Transformer blocks so we removed the classification token in the input. As all move features are fed into the Transformer simultaneously, we will need to retain positional information of the chess moves by adding a positional encoding to each move feature. ViT adds a 1D learnable position embedding to the input sequence but it is not flexible when it comes to variable length of input during training. Therefore, we use sine and cosine functions to encode positions, which was proposed in the standard Transformer, and support up to 500 chess moves as the maximum length.

As an overview of our model, each input move feature will be firstly projected to 1024 dimensions and positional encoding will be added to the resulting vectors before they are fed into the Transformer encoder. The encoder has 12 Transformer blocks, and each is composed of an attention layer followed by a feed-forward layer, and layer normalization (Ba et al., 2016) is applied before each layer. There are 8 attention heads, each with 64 dimensions in each attention layer. The feed-forward layer uses MLP with two layers that takes in a 1024-dimensional vector which will be projected to 2048 dimensions and back to 1024 dimensions. GELU activation was used in between the two layers. Each output vector of the encoder will correspond to each input move feature, and we average these vectors to get a single vector of dimension 1024, which will then be projected to our final embedding size of 512 with MLP. Tanh activation is applied in the end. Thus, we finally have our a 512-dimensional embedding vector that encodes the decisions the player made in the game. At inference time, the player vector will be obtained by simply averaging all the game vectors (i.e. the centroid) that belong to the player.

Loss Function. In order to capture individual playing style, we adapt *Generalized End-to-End Loss for Speaker Verification (GE2E)* (Wan et al., 2018) to our setting, which allows us to train the game vectors to maximize both intra-player compactness and inter-player discrepancy in the resulting embedding space. GE2E builds a similarity matrix on a batch of $N \times M$ games, where N is number of players and M is number of games per player. For each game from each player, we extract the current 512-dimensional embedding vector $\mathbf{e}_{ji}$ (i-th game from the j-th player) output by the residual blocks and Transformer network. We also find the centroid vector $\mathbf{c}_j$ of a specific target player by averaging all of their game vectors in the batch. However, we remove $\mathbf{e}_{ji}$ when computing the centroid of the true player as stated in Wan et al. (2018) to help avoid trivial solutions.

$$\mathbf{c}_j^{(-i)} = \frac{1}{M-1} \sum_{\substack{m=1 \\ m \neq i}}^{M} \mathbf{e}_{jm}, \tag{1}$$

A similarity matrix $\mathbf{S}_{ji,k}$ will then be constructed with the cosine similarity between each embedding vector and the centroid of all players present in the batch, and will be further scaled by two learnable parameters w and b. We will simply use the average of game vectors as centroid when calculating the negative similarity ($k \neq j$) and use Equation 1 to find the centroid otherwise.

$$\mathbf{S}_{ji,k} = \begin{cases} w \cdot \cos(\mathbf{e}_{ji}, \mathbf{c}_j^{(-i)}) + b & \text{if} \quad k = j; \\ w \cdot \cos(\mathbf{e}_{ji}, \mathbf{c}_k) + b & \text{otherwise.} \end{cases} \tag{2}$$

The final loss function will be trying to maximize the similarity for the correct entries while penalizing all other entries in the similarity matrix. Therefore, the loss function is pushing each embedding vector closer towards the player's centroid and further from all other players' centroids simultaneously. Finally, we can sum up all the losses from each embedding vector in the similarity matrix.

$$L(\mathbf{e}_{ji}) = -\mathbf{S}_{ji,j} + \log \sum_{k=1}^{N} \exp(\mathbf{S}_{ji,k}), \qquad L_{GE2E} = \sum_{j,i} L(\mathbf{e}_{ji}) \tag{3}$$

Training procedure. We randomly sample $N \times M$ games as a batch, where $N = 40$ is number of players and $M = 20$ is number of games per player. In order to speed up training and perform batch computing, we randomly sample a 32-move sequence from each game and pad 0s to games with length less than 32 moves. The entire model was trained with 4 Tesla K80 GPUs and SGD optimizer with an initial learning rate of 0.01 and momentum of 0.9. The learning rate was reduced by half every 40K steps. Initial values of w and b for the similarity matrix were chosen to be $(w, b) = (10, -5)$, and we used a smaller gradient scale of 0.01 to match the original GE2E model.

4 Evaluation

In this section, we apply our methodology to train several models and evaluate their performance. We first describe the datasets we use and a competitive baseline, then compare our approach with the baseline on a set of behavioral stylometry tasks.

4.1 Data

Lichess. We base our dataset of chess games on the one used by McIlroy-Young et al. (2020b). Our main set of games is derived from Lichess, a popular open-source chess platform. Their creative commons licensed database has over two billion games played by players ranging from rank beginners to the current world champion, Magnus Carlsen. From this database, we collect rated Blitz games (3–7 minutes per player per game) played on Lichess between January 2013 and December 2020 by players who match all of the following criteria: active as of December 2020, mean rating between 1000–2000, low variance in Blitz rating, and at least 1,000 games played during the time interval. We also restrict our attention to games with at least 10 moves (20 ply). To investigate whether behavioral stylometry performance varies with number of games played, we bucket the players according to how many games they have played: 1K–5K, 5K–10K, 10K–20K, 20K–30K, 30K–40K, and 40K+ (buckets are hereafter referred to by the minimum of their respective range, e.g. "20K" for the 20K–30K bucket). Players are further divided into *learning* and *evaluation* sets, using the same players as in (McIlroy-Young et al., 2020b)[1] for the 10K to 40K players. The learning sets of players are used to train the models, and evaluation players, unseen during training, are used to perform few-shot classification. Finally, each player's games are randomly split into a training set (80%), a validation set (10%), and a test set (10%). In all of our evaluations, we only use 200 games per player, rather than the full test set. The numbers of players and games in each player set is shown in Table 1.

High-ranking players. We also assembled a dataset of high-ranking online players from both Lichess and Chess.com, the two most popular online chess platforms. To do so, we took the lists of the top 1,500 players on both sites' leaderboards, downloaded all of their games, and filtered out

[1] We obtained the list of players via private communication.

Dataset	1K	5K	10K	20K	30K	40K+	Ranked
Learning Players	8,617	5,298	1,866	295	86	19	1,813
Learning Games	14M	24M	17M	4.9M	2.0M	0.7M	9.5M
Evaluation Players	23,101	1,324	466	72	20	20	451
Evaluation Games	4.6M	264.8K	93.2K	14.4K	4K	4K	90.2K

Table 1: The numbers of players and games used in training (*learning*) and testing (*evaluation*) sets.

games with fewer than 10 moves and players with fewer than 950 games. We split the remaining high-ranking players into *learning* (80%) and *evaluation* (20%) sets, and their counts are shown in Table 1 (again, only 200 test games per player are used in our evaluations). Note that these games include all time controls, tend towards faster time controls than the general population. These sets include many of the best players in the world, including the world champion.

4.2 Experimental Setup

Task formulation. Our central evaluation task is performing behavioral stylometry in chess, in which the goal is to identify a target player from amongst a pool of potential players D given only the moves they played in a particular set of games. We set this up as a few-shot classification task, in which for each player in a set of players R, unseen during training, we are given a small, labeled set of support games of size r_s. Then, for each $r \in R$, we are given another small set of query games, this time unlabeled and of size r_q, and we seek to identify which player in D played them. We call D the *domain* set and R the *range* set. $R \subseteq D$, and in general D can be much larger than R—for example, one possible task would be to try to identify some set of $|R| = 10$ unknown players, but we are only promised that they are members of some much larger set of $|D| = 10,000$ players. Finally, we also specify a task parameter k where all games seen, both labeled and unlabeled, only contain moves starting at move k. As we will see, the opening moves in chess can be highly discriminative because players repeatedly play the same openings. For $k = 0$ we are given entire games, but for e.g. $k = 15$, we are only given moves 15 and beyond for each game. The behavioral stylometry task is thus fully specified by D, R, r_s, r_q, and k: for each player in D we are given r_s labeled games and for each player in R we are given r_q unlabeled games, all starting at move k. For each set of r_q unlabeled games we seek to identify which player in D played them.

Openings baseline. Chess games always start from the same initial position, and the opening moves are heavily studied and well-understood. Opening choice strongly influences the type of resulting game, and thus players often study a small set of openings in depth and stick to them. As a result, the opening moves are likely to contain a lot of information about who is playing the game, albeit for the somewhat unsatisfying reason that they are memorized patterns of play rather than genuine decisions being made. Indeed, when strong anonymous accounts play on online platforms, informed observers often rely on their opening moves to guess who they are. As a strong baseline for comparison, we create an embedding model that uses the opening moves. The model creates *game vectors* by sampling 5 sequential moves by the target player, converts each into a one-hot 4096-dimensional encoding vector (moves that are never used will be 0 for all vectors and do not effect our distance measurements), and adds them together to yield a 5-hot vector encoding of the 5 opening moves. Doing this process for each of the r_q games in the target player's set of unlabeled games yields r_q 5-hot vectors in a 4096-dimensional space, which we then convert into a player vector by simply taking their centroid. These player vectors can then be used in the same manner as those in our main model to perform behavioral stylometry from the opening moves.

Our model. We apply our methodology to the behavioral stylometry task. To construct our main model, we trained on all training games in the learning sets of the Lichess players (1k –40k+ buckets) — 63.7M games by 16,181 players in total. Our model learns player vectors in an embedding space that captures chess style into which new, unseen players can be encoded. To perform few-shot stylometry on an unseen player p with domain D, we first infer labeled player representations from the support set of each player in D, then infer p's unlabeled representation by running the model on p's query set. We then compute the cosine distance between p's query representation and all

domain players' support representations and return the closest domain player as p's identity. In all of our evaluations, the support sets are drawn from players' 10% validation sets and the query sets are drawn from players' 10% testing sets. We report precision at 1 (P@1) unless otherwise noted. We also trained two additional models, one on high-ranking players and one on the set of players used in McIlroy-Young et al. (2020b), which are discussed in the Supplement.

4.3 Behavioral Stylometry

Stylometry performance. Our main evaluation is the behavioral stylometry task where $D =$ 10K $\cup$ 20K $\cup$ 30K $\cup$ 40K+ (all 2,844 Lichess players with at least 10,000 games played, in both the learning and evaluation sets, are possible identities), $R =$ 10K (the 578 Lichess players in the 10K evaluation set are the target players to be identified), $n_s = n_q = 100$ (support and query sets are 100 games each in size), and $k = 15$ (we only get games starting from move 15, so that memorized openings do not factor into our evaluation of capturing human decision-making style). On this task, our model achieves a Precision@1 of 0.86, exactly identifying a player from 100 of their games out of a possible 2,844 players 86% of the time (see Table 2(top)). Furthermore, this is without access to the opening, the most regimented and repeated part of the domain. In comparison, the baseline with access to the first 5 given moves of each game (i.e. moves 16–20) only achieves a Precision@1 score of 0.234—far better than random guessing, indicating that exact moves indeed capture a significant amount of information, but far worse than our method. In the Supplement, we also show how performance varies for different cutoffs of minimum number of games per player. Both our model and the baseline are robust to this choice, indicating that the task does not depend on players having played many games.

A powerful property of our method is that it adapts to players it has never seen before. We compare performance on unseen players (i.e. those in the evaluation sets) versus seen players (i.e. those in the learning sets that were used during model training). The difference in performance between players the model explicitly trained on and players it has never seen before is surprisingly small. For example, the model achieves P@$1 = 0.86$ on unseen players and P@$1 = 0.853$ on seen players. This suggests that the model has learned a comprehensive space of decision-making style.

In Table 2(bottom), we also show the version of the task where the opening moves are given to the models ($k = 0$). Our model performance increases to near-perfect accuracy, but the openings baseline also increases substantially. This indicates that using the opening moves to perform stylometry—identical moves that players repeat out of memorization—is less interesting than using moves later on during games, when players are making decisions in instances they (and our models) haven't seen before. Although our model is still significantly better than the openings baseline even in this formulation of the task, we use $k = 15$ for the rest of our analyses.

Comparison with personalized move-matching models. The previous state of the art performance for chess player stylometry was achieved by training individual move-matching models for every player in D (McIlroy-Young et al., 2020b). This requires thousands of labeled games per player, and is only computationally feasible for small $|D|$, hence we are unable to include this method in our main analysis due to its lack of scalability. However, we directly compare our method on the small, 400-player domain tested in McIlroy-Young et al. (2020b). For this comparison, the domain and range are both 400 players, $k = 15$, and $r_s = r_q = 100$. The results are shown in Table 3(top), where personalized models achieve Precision@1 of 0.552. However, our model achieves Precision@1 of 0.953, far outstripping the previous method's performance.

High-ranking players. To further study how our model performs on out of sample examples, we tested on our set of high-ranking players (Section 4.1). These players are not only unseen by the model, but they are of much higher skill than any of the players it was trained on, and half of them are drawn from a separate online platform. We begin with the task of identifying the 408 evaluation high-ranked players out of the complete set of high-ranked players (Table 3(bottom)). This task is significantly more difficult, as both the baseline and our model have lower P@1 scores. However, our model still outperforms the baseline by almost a factor of 10. Interestingly, when the set of Lichess amateurs is added to the domain D, the model performance hardly changes at all, indicating that high-ranked players occupy a different part of the embedding space.

Range (R)	$\|R\|$	$\|D\|$	k	Random	Baseline	Our Model
10k+ Lichess (unseen)	578	2844	15	0.0004	0.244	**0.860**
10k+ Lichess (seen)	2266	2844	15	0.0004	0.274	**0.853**
10k+ Lichess (all)	2844	2844	15	0.0004	0.268	**0.854**
10k+ Lichess (unseen)	578	2844	0	0.0004	0.929	**0.979**
10k+ Lichess (seen)	2266	2844	0	0.0004	0.930	**0.982**
10k+ Lichess (all)	2844	2844	0	0.0004	0.929	**0.982**

Table 2: Behavioral stylometry performance on Lichess players with all games starting at move $k = 15$ (top) or $k = 0$ (bottom). The domain is the entire set of Lichess players with over 10K games, and the range is all players with over 10K games in either the evaluation set (unseen), learning set (seen), or both (all). 100 games were used for both support and query sets of each player, drawn from games in players' validation and test sets.

Task	$\|R\|$	$\|D\|$	Baseline	Personalized	Our Model
McIlroy-Young et al.	400	400	0.478	0.552	**0.953**
High-ranked players	408	2157	0.039	—	**0.308**
" $\cup$ 10K–40K+ in D	408	5001	0.027	—	**0.301**

Table 3: (Top) Comparison with personalized move-prediction models in McIlroy-Young et al. (2020b) with $k = 15$. (Bottom) Performance on high-ranked players, either with high-ranked players only in the domain or high-ranked players and Lichess amateurs together, with $k = 15$ and $n_s = n_q = 100$.

Support and query set sizes. Here we show how model performance varies with r_s and r_q. As Figure 2a shows, model performance rises sharply for early values and starts to saturate around $r_s = r_q = 50$ games. Even when given very small numbers of labeled examples, i.e. $r_s = 10$, our model performs well. In comparison, personalized move-prediction models require thousands of games to begin seeing improvements over unpersonalized models (McIlroy-Young et al., 2020b).

Expanded dataset. We examined performance on our complete dataset (1K to 40K+), comprising 41,184 players for both the domain and range. Even in this extensive regime, we achieve 54.0% P@1 while the baseline model achieves 8.49% P@1 (see Supplement for details).

4.4 Qualitative Analysis

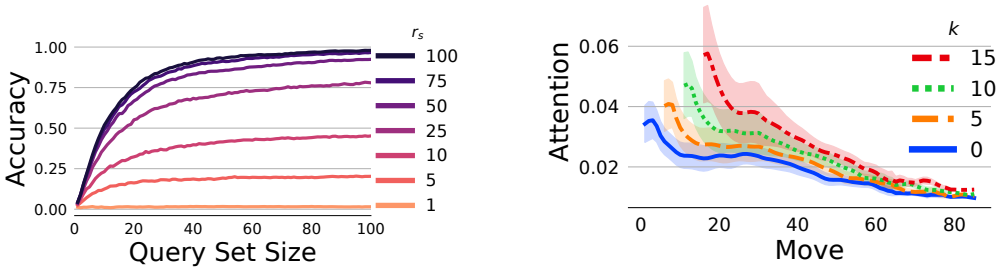
Attention. Our use of attention allows us to examine which sections of the game are most useful for identifying individual style. Figure 2b shows the average per-move attention for the main model when applied to 100 games for each player in the 40K+ set. Four different starting moves were examined ($k = 0, 5, 10, 15$) to show how the models handle being given incomplete games. The earlier parts of the game are most informative, with a monotonic decay in attention as games progress.

Playing style. Figure 3 shows three different colorings of a 2-D t-SNE projection of the high skill players[2]. The colorings are based on the preferred opening moves of the players, either when playing as White (Figure 3a), or when playing as Black and responding to the two most common opening moves, e4 (Figure 3a and d4 (Figure 3a). Despite these player vectors being learned from games starting on move $k = 15$, where the opening moves were discarded, the resulting player embeddings are tightly related to their opening preferences. This indicates that the space is aligned with the stylistic preferences of players.

5 Conclusion

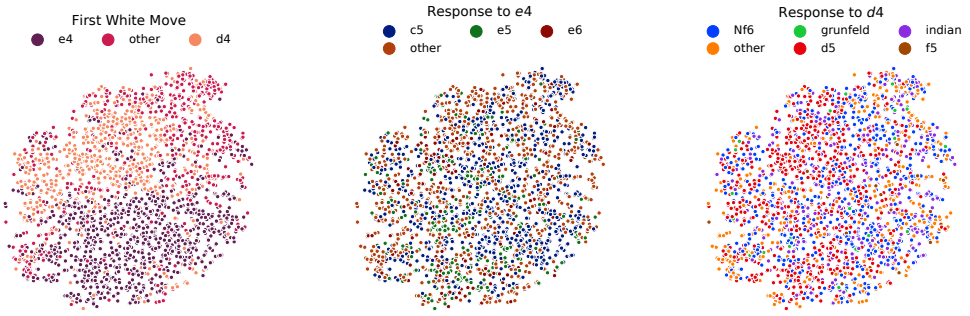
Motivated by the issues inherent in algorithmically characterizing human behavior, we propose a method for performing behavioral stylometry – identifying people down to the level of specific

[2]See supplement for instructions on viewing them interactively



(a) (b)

Figure 2: (a) Stylometry performance (P@1) as a function of query set size (r_q) and support set size (r_s). (b) Distribution of attention across 1,900 games (40K+ learning players) from 4 different starting moves (k). Confidence intervals represent 1 standard deviation.



(a) Colored by first move as white.

(b) Colored by first move in response to $e4$.

(c) Colored by first move in response to $d4$.

Figure 3: Two-dimension t-SNE projection of high-ranked player vectors generated from games starting at move $k = 15$.

individuals from their patterns of decisions – and demonstrate the effectiveness of these methods in the domain of chess. We adapt techniques developed for speaker verification tasks to this decision-making setting, and construct a neural network architecture that can reliably identify individual-level decision-making style. Our architecture uses residual blocks to extract position- and move-level features, then combines these with a position encoding and inputs them to a Vision Transformer that learns how to accurately aggregate this information into a game representation. We use a GE2E loss framework to maximize both intra-player compactness and inter-player discrepancy in the resulting embedding space. The final model is then capable of performing highly accurate few-shot classification of unseen players, out-performing both the previous state of the art and a competitive opening baseline. The resulting embedding space generalizes in several ways—it can identify high-ranked players even though it was only trained on amateurs, it identifies unseen players equally well as players it was trained on, and maintains its strong performance when expanded to tens of thousands of players.

Our methodology has several limitations that raise natural questions for future work. First, while our methodology far out-performs the baseline on top players, the worse performance suggests there is room to improve in capturing high-level, complex styles. Second, our methodology is geared towards identifying individuals from their behavior alone, which could lead to privacy concerns. While this critique pertains to any stylometry task, including hand-writing recognition and speaker verification, our work raises similar questions in behavioral domains. Future work could investigate whether the benefits of methodologies capable of stylometry could be achieved without the privacy risks of uniquely identifying people. Finally, our methodology could be extended to other behavioral domains where fruitful human-AI collaboration is on the horizon.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We discuss them in the Conclusion.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We discuss the privacy risks in the Conclusion.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Our supplement contains the code, the model files are too large for the supplement so will be released when the paper goes public
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] We have data and methods sections discussing this along with more information in the supplement
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] We ran over thousands of players to get error estimates, but did not train or test multiple times
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Our methods and supplement discuss this
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] Our code release links to the code we build on
   (b) Did you mention the license of the assets? [Yes] The code we are using is GPL or MIT licensed and is mentioned in the code release
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We plan to release the code and models after publication
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We discuss this in the Data section, all data used is public and not sensitive
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] We discuss this in the Data section, there is no PII in our data

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

References

Ashton Anderson, Jon Kleinberg, and Sendhil Mullainathan. 2017. Assessing human error against a benchmark of perfection. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11, 4 (2017), 45.

Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. 2016. Learning to learn by gradient descent by gradient descent. In *Advances in neural information processing systems*. 3981–3989.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer Normalization. arXiv:1607.06450 [stat.ML]

Philip Bachman, Alessandro Sordoni, and Adam Trischler. 2017. Learning Algorithms for Active Learning. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 301–310. `http://proceedings.mlr.press/v70/bachman17a.html`

A. Bakhtin, S. Gross, Myle Ott, Y. Deng, Marc'Aurelio Ranzato, and Arthur D. Szlam. 2019. Real or Fake? Learning to Discriminate Machine from Human Generated Text. *ArXiv* abs/1906.03351 (2019).

Tamal Biswas and Kenneth W Regan. 2015. Measuring Level-K Reasoning, Satisficing, and Human Error in Game-Play Data. In *2015 IEEE 14th International Conference on Machine Learning and Applications*. IEEE, Miami, FL, 941–947.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems* 6 (1993), 737–744.

Aylin Caliskan-Islam, Richard Harang, Andrew Liu, Arvind Narayanan, Clare Voss, Fabian Yamaguchi, and Rachel Greenstadt. 2015. De-anonymizing Programmers via Code Stylometry. In *24th USENIX Security Symposium (USENIX Security 15)*. USENIX Association, Washington, D.C., 255–270. `https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/caliskan-islam`

Neil Charness. 1992. The impact of chess research on cognitive science. *Psychological research* 54, 1 (1992), 4–9.

Jonghyun Choi, J. Krishnamurthy, Aniruddha Kembhavi, and Ali Farhadi. 2018. Structured Set Matching Networks for One-Shot Part Labeling. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), 3627–3636.

S. Chopra, R. Hadsell, and Y. LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 1. 539–546 vol. 1. `https://doi.org/10.1109/CVPR.2005.202`

Joon Son Chung, Jaesung Huh, Seongkyu Mun, Minjae Lee, Hee-Soo Heo, Soyeon Choe, Chiheon Ham, Sunghwan Jung, Bong-Jin Lee, and Icksang Han. 2020. In Defence of Metric Learning for Speaker Recognition. In *Proc. Interspeech 2020*. 2977–2981. `https://doi.org/10.21437/Interspeech.2020-1064`

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. 2019. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 4685–4694. `https://doi.org/10.1109/CVPR.2019.00482`

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. `https://doi.org/10.18653/v1/N19-1423`

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*. 647–655.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*. https://openreview.net/forum?id=YicbFdNTTy

Li Fei-Fei, R. Fergus, and P. Perona. 2006. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28, 4 (2006), 594–611. https://doi.org/10.1109/TPAMI.2006.79

Michael Fink. 2005. Object Classification from a Single Example Utilizing Class Relevance Metrics. In *Advances in Neural Information Processing Systems*, L. Saul, Y. Weiss, and L. Bottou (Eds.), Vol. 17. MIT Press. https://proceedings.neurips.cc/paper/2004/file/ef1e491a766ce3127556063d49bc2f98-Paper.pdf

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 1126–1135.

Andrew Gibiansky, Sercan Arik, Gregory Diamos, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou. 2017. Deep Voice 2: Multi-Speaker Neural Text-to-Speech. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/c59b469d724f7919b7d35514184fdc0f-Paper.pdf

Sumit Goswami, Sudeshna Sarkar, and Mayur Rustagi. 2009. Stylometric Analysis of Bloggers' Age and Gender. In *ICWSM*.

Luiz G. Hafemann, Robert Sabourin, and Luiz S. Oliveira. 2016. Writer-independent feature learning for Offline Signature Verification using Deep Convolutional Neural Networks. In *2016 International Joint Conference on Neural Networks (IJCNN)*. 2576–2583. https://doi.org/10.1109/IJCNN.2016.7727521

Luiz G. Hafemann, Robert Sabourin, and Luiz S. Oliveira. 2017. Learning Features for Offline Handwritten Signature Verification Using Deep Convolutional Neural Networks. 70, C (2017). https://doi.org/10.1016/j.patcog.2017.05.012

Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks, In Proceedings of the IEEE conference on computer vision and pattern recognition. *arXiv*, 7132–7141.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. In *Proceedings of the 22nd ACM International Conference on Multimedia* (Orlando, Florida, USA) *(MM '14)*. Association for Computing Machinery, New York, NY, USA, 675–678. https://doi.org/10.1145/2647868.2654889

Ye Jia, Yu Zhang, Ron J Weiss, Quan Wang, Jonathan Shen, Fei Ren, Zhifeng Chen, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, et al. 2018. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. *arXiv preprint arXiv:1806.04558* (2018).

Chao Li, Xiaokong Ma, Bing Jiang, Xiangang Li, Xuewei Zhang, Xiao Liu, Ying Cao, Ajay Kannan, and Zhenyao Zhu. 2017. Deep Speaker: an End-to-End Neural Speaker Embedding System. (05 2017).

Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. 2017. SphereFace: Deep Hypersphere Embedding for Face Recognition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 6738–6746. https://doi.org/10.1109/CVPR.2017.713

Reid McIlroy-Young, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. 2020a. Aligning Superhuman AI with Human Behavior: Chess as a Model System. In *Proceedings of the 25th ACM SIGKDD international conference on Knowledge discovery and data mining*.

Reid McIlroy-Young, Russell Wang, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. 2020b. Learning Personalized Models of Human Behavior in Chess. *arXiv preprint arXiv:2008.10086* (2020).

Tempestt Neal, Kalaivani Sundararajan, Aneez Fatima, Yiming Yan, Yingfei Xiang, and Damon Woodard. 2017. Surveying Stylometry Techniques and Applications. *ACM Comput. Surv.* 50, 6, Article 86 (Nov. 2017), 36 pages. https://doi.org/10.1145/3132039

Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. 2014. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1717–1724.

Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O. Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. 2018. Deep Voice 3: 2000-Speaker Neural Text-to-Speech. In *International Conference on Learning Representations*. https://openreview.net/forum?id=HJtEm4p6Z

Sachin Ravi and Hugo Larochelle. 2017. Optimization as a Model for Few-Shot Learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=rJY0-Kcll

Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. 2018. Meta-Learning for Semi-Supervised Few-Shot Classification. In *Proceedings of 6th International Conference on Learning Representations ICLR*.

Hanan Rosemarin and Ariel Rosenfeld. 2019. Playing Chess at a Human Desired Level and Style. In *Proceedings of the 7th International Conference on Human-Agent Interaction*. 76–80.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 815–823. https://doi.org/10.1109/CVPR.2015.7298682

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. 2018. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* (2018).

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *Nature* (2017).

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical Networks for Few-shot Learning. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/cb8da6767461f2812ae4290eac7cbc42-Paper.pdf

David Snyder, Pegah Ghahremani, Daniel Povey, Daniel Garcia-Romero, Yishay Carmiel, and Sanjeev Khudanpur. 2016. Deep neural network-based speaker embeddings for end-to-end speaker verification. In *2016 IEEE Spoken Language Technology Workshop (SLT)*. 165–170. https://doi.org/10.1109/SLT.2016.7846260

Nima Tajbakhsh et al. 2016. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE transactions on medical imaging* 35, 5 (2016), 1299–1312.

Sebastian Thrun. 1998. *Lifelong Learning Algorithms*. Springer US, Boston, MA, 181–209. https://doi.org/10.1007/978-1-4615-5529-2_8

Fiona J. Tweedie, Sameer Singh, and David I. Holmes. 1996. Neural network applications in stylometry: The *Federalist Papers*. *Comput. Humanit.* 30, 1 (1996), 1–10. https://doi.org/10.1007/BF00054024

Ehsan Variani, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez. 2014. Deep neural networks for small footprint text-dependent speaker verification. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 4052–4056. https://doi.org/10.1109/ICASSP.2014.6854363

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. 2016. Matching Networks for One Shot Learning. In *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Eds.), Vol. 29. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2016/file/90e1357833654983612fb05e3ec9148c-Paper.pdf

Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. 2018. Generalized End-to-End Loss for Speaker Verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 4879–4883. https://doi.org/10.1109/ICASSP.2018.8462665

Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. 2018. CosFace: Large Margin Cosine Loss for Deep Face Recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5265–5274. https://doi.org/10.1109/CVPR.2018.00552

Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. 2020. Generalizing from a Few Examples: A Survey on Few-Shot Learning. *ACM Comput. Surv.* 53, 3, Article 63 (June 2020), 34 pages. https://doi.org/10.1145/3386252

Shi-Xiong Zhang, Zhuo Chen, Yong Zhao, Jinyu Li, and Yifan Gong. 2016. End-to-End attention based text-dependent speaker verification. In *2016 IEEE Spoken Language Technology Workshop (SLT)*. 171–178. https://doi.org/10.1109/SLT.2016.7846261

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. *arXiv* (2016).