MDPI

Article

Meta Learning Based Deception Detection From Speech

Noa Mansbach, Amos Azaria

Computer Science Department, Ariel University, Ariel, Israel
noa.aizer@msmail.ariel.ac.il, amos.azaria@ariel.ac.il

Abstract: It is difficult to overestimate the importance of detecting human deception, specifically by using speech cues. Indeed, several works attempt to detect deception from speech. Unfortunately, most works use the same people and environments in training and in testing. That is, they do not separate training samples from test samples according to the people who said each statement or by the environments in which each sample was recorded. This may result in less reliable detection results. In this paper, we take a meta-learning approach in which a model is trained on a variety of learning tasks to enable it to solve new learning tasks using only a few samples. In our approach, we split the data according to the persons (and recording environment), i.e., some people are used for training, and others are used for testing only, but we do assume a few labeled samples for each person in the data set. We introduce CHAML, a novel deep learning architecture that receives as input the sample in question along with two more truthful samples and non-truthful samples from the same person. We show that our method outperforms other state-of-the-art methods of deception detection based on speech and other approaches for meta-learning on our data-set. Namely, CHAML reaches an accuracy of 61.34% and an F1-Score of 0.3857, compared to an accuracy of only 55.82% and an F1-score of only 0.3444, achieved by a previous, most recent approach.

Keywords: Deception detection; Speech classification; Meta-learning

1. Introduction

Lying and deception are an inherent part of human nature. While some lies are considered small and may even be helpful for a smoother interaction between humans, others may be devastating and cause major damage. However, despite deception detection being essential to everyone in their daily life, it is challenging for humans to determine whether a person is being deceptive [1,2]. Therefore, throughout history, many methods and devices were developed for that task [3], and more recently, machine learning methods based on text, video, and speech [4–6].

Typically, works for speech deception detection do not separate train and test based on the person so that an individual may have examples in both train and test [5]. Nor do most works split the training and test data according to the recording environment. This results in less reliable models, as in practice, the model must learn about some population in some recording environment and then be used to produce predictions for a different population in a different recording environment. Consequently, in this work, we split the data according to the persons, i.e., some people are used for training, and others are used for testing only.

One of the main difficulties in deception detection based on speech is to learn the features of lying from some people and in some recording environments and apply this knowledge to others, despite the fact that different people tend to lie differently. Therefore, in order to achieve high performance, the model is required to learn which speech-related features are specific to a person and which are general.

In addition, we consider a meta-learning approach for deception detection based on speech. For that, we assume that we have very few labeled samples for each person (namely, two positive samples and two negative ones). This approach is inspired by the

Citation: Mansbach, N.; Azaria, A. Meta Learning Based Deception Detection From Speech. *Appl. Sci.* **2022**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

well-known polygraph test [7], in which several comparison questions are asked at the beginning of the polygraph interview in order to obtain physiological measures of the subject when telling the truth and when lying. Namely, in our approach, the model is trained on a set of training tasks; each task represents a person from the subjects in our data set and consists of a support set used for learning about the task and a query set used to evaluate the performance of this task. The support set contains four examples from the person's samples, two positive samples and two negative ones, and the query set contains the remaining samples of the person.

Our approach to meta-learning differs from the typical one. In a typical meta-learning problem, there are typically several classes in each task, which differ from task to task. In addition, training and test tasks typically have different classes. However, in our setting, we have the same two classes for all tasks, but each task represents a different person from our data, and therefore, the data features are very different between different tasks. We present an innovative method of deception detection in the meta-learning setting and show that it outperforms the existing state-of-the-art methods of deception detection based on speech and other approaches for meta-learning. In our method, we use the comparative hint approach, which gives the model hints about the new environment (in our case, a new person) by providing some true and false examples from the same person sample set together with the tested sample. Namely, our model processes the positive and negative pairs and combines the result of this process with the tested sample into a vector fed into a neural classifier.

We believe that by using our unique architecture, the model can compare the tested sample to the given hints, learn the person's way of lying, and improve its detection performance.

To summarize, our main contribution in this work is tackling the problem of deception detection based on audio signals when having different train and test environments (i.e., persons), and when the model is provided with very few true and false labeled samples for each person in the test-set. To that end, we gathered a massive amount of data and developed CHAML, a novel solution based on the meta-learning approach that uses samples for each person to learn their way of lying. This method outperforms state-of-the-art methods and can be applied to other environments that include different tasks, using any neural network as its classifier.

2. Related Work

2.1. Deception Detection

The deception detection task has been explored in many types of research using different approaches and techniques: some based on text, some on speech, some on video, and others on physiological measures. In the beginning, most research was focused on analyzing physiological measures, such as breathing rate, heart rate, blood pressure, and body temperature [8–10]. Other studies found the connection between deception and human behaviors [11–13].

However, detecting the physiological measures of a human requires special instruments and may be invasive and expensive [13,14]. Therefore,many studies researched the use of machine learning methods for deception detection.

For detection based on text problems, Ott et al. [4] develop a corpus of deceptive reviews and use Näive Bayes and Support Vector Machine (SVM) as the classifiers utilizing Linguistic Inquiry and Word Count (LIWC) combined with bigrams. Feng et al. [15] show that using Context Free Grammar (CFG) parse trees consistently improves detection performance. Barsever et al. [16] use the BERT (Bidirectional Encoder Representations from Transformers) network and show that compared with truthful text, deceptive text tends to be more formulaic and less varied.

There have been only a few works that attempt to detect deception based on speech cues. Hirschberg et al. [17] develop a corpus of deceptive speech using one on one interviews. Nasri et al.[18] use SVM model utilizing Mel Frequency Cepstral Coefficient

(MFCC). The MFCC is a feature representation commonly used in speech processing and speech recognition tasks. MFCCs are derived from the spectral representation of a speech signal and are used to capture the spectral characteristics of the signal, such as the frequencies of the various speech sounds and the power spectrum of the signal. MFCCs are commonly used for speech analysis in various domains, such as voice recognition [19], speech recognition [20], emotion recognition [21], animal vocalizations [22], and neonatal bowel sound detection [23,24].

Graciarena et al. [25] train a classifier using combined both linguistic features and acoustic features as a combination of MFCC and prosodic features and Marcolla et al. [26] use an LSTM neural network on a set of MFCC characteristics extracted from audio speech to deception detecting based on voice stress. Xie et al. [27] extract variable length frame-level speech features from different length's speech samples and use a recurrent neural network combined with a convolution operation as their model.

Other studies focus on deception detection in videos using different feature extraction methods such as IDT (Improved Dense Trajectory) feature, and high-level features represent facial micro-expressions extracted from the videos, with machine learning techniques [28–31]. Ding et al.[28] develop an automated deception detection model that consists of three main modules: face focused cross-stream network which deep joint feature learning from facial expressions and body motions for video, a meta-learning module, and an adversarial learning module that generates a 256-dimension feature vector for each synthesized video. The meta-learning module was used to deal with their data scarcity problem by using pairwise comparison. Each deceptive video sample was combined with four true samples to generate five pairs. The model outputs the probability for each pair to be from the same class.

Other researchers apply a multi-modal approach for deception detection from video data sets. Pérez-Rosas et al. [6] introduce a collected data-set consisting of videos collected from public court trials. They apply a multi-modal approach for deception detection on their data set using inputs from different modalities, i.e., video, audio, and text. Wu et al.[32] use common machine learning techniques such as SVM, Näive Bayes, Decision Trees, Random Forests, Logistic Regression, and Adaboost. They test different combinations of multi-modal features: IDT feature and high-level features represent facial micro-expressions extracted from the videos as their motion features, MFCC as the audio features and encoded the video transcripts using Glove (Global Vectors for Word Representation). Other researches [33,34] introduce a deep learning multi-modal approach on the same data-set, using Multi-Layer Perceptron (MLP) and Convolutional Neural Network (CNN) models. More specifically, Gogate et al. [33] present a deep CNN approach that utilize both early and late multi-modal fusion methods, incorporating audio, visual, and textual features. In the early fusion approach, audio, visual, and textual features are extracted using an openSMILE feature extractor, a 3D-CNN, and a CNN applied to the GloVe embedding of the video transcript, respectively. These features are then concatenated and input into an MLP classifier. In the late fusion approach, separate unimodal classifiers are trained to obtain predicted labels, concatenated, and fed into an MLP classifier to obtain the final predicted label. Krishnamurthy et al. [34] propose a similar approach using an MLP model that takes a multi-modal feature representation of a video as input. This representation includes visual features extracted using a 3D CNN, textual features extracted using a CNN applied to the Word2Vec embedding of the video transcript, audio features extracted using the openSMILE toolkit, and micro-expression features derived from ground truth annotations. The authors use two data fusion techniques for combining these features; concatenation and Hadamard product followed by concatenation. Both papers achieve an accuracy of 96% on the [6] data-set with their multi-modal model, outperforming other baseline methods that only considered visual and textual features. However, as noted by the authors, their performance may not extend to larger data-sets or out-of-domain scenarios.

2.2. Speech Emotion Recognition

The speech classification task has many fields such as emotion recognition, speaker identification, language identification, etc. The Speech Emotion Recognition (SER) task is recognizing the emotional aspects of speech and classifying them into emotion categories. This task may be considered very close to our task, as an emotional aspect may be involved in people's lying.

Many previous works on the SER problem proposed solutions based on classical ML methods such as Hidden Markov Models (HMM), SVM, and Random Forests [35–37]. However, in the past several years, deep learning has become one of the main approaches for solving the SER problem [38]. Trigeorgis et al. [39] develop an end-to-end speech emotion recognition using a combination of CNN with LSTM networks for learning a representation of the speech signal directly from the raw time representation. Han et al.[40] introduce a method that uses a neural network for extracting high-level features from audio and producing an emotion state probability distribution for each speech segment. Fayek et al. [41] propose a deep-learning framework using CNNs and a spectrogram of a speech signal as the input. Another promising approach used in various speech-related tasks is the Wav2Vec 2.0 framework [42]. This framework is used for self-supervised learning of vector representation from speech audio. Recent research has shown that the Wav2Vec 2.0 framework is also a robust alternative for SER and speaker identification tasks [43–45]. Therefore, we also use this framework for one of the extraction feature types in our task.

2.3. Few-Shot Learning

The method in our paper is based on the idea of the meta-learning framework or "learning to learn" [46], specifically in the field of few-shot learning. The meta-learning framework is based on learning from prior experience with other tasks, i.e., learning how to learn to classify given a set of training tasks such that the model can solve new learning tasks. One of the main challenges in the meta-learning framework is to train an accurate model using only a few training examples given prior experience with similar tasks. This is called few-shot learning.

Few-shot learning, is training a model for learning from very few samples and generalizing to many other new examples; most approaches for few-shot learning follow the meta-learning framework. It measures a model's ability to quickly adapt to new environments and tasks using only a few examples and training iterations. For that, the model is trained on a set of tasks in a meta-learning phase, allowing it to adapt quickly to new tasks with just a few examples. Each task consists of a support set used for learning how to solve each specific task and a query set containing further examples of each specific task, which are used to evaluate the performance on each task. A task can be utterly non-overlapping with another; the classes from one task may never appear in another. The model's performance is evaluated by the average test accuracy across the query sets of many testing tasks. As the meta-learning process proceeds, the model parameters are updated based on the training tasks. The loss function is derived from the classification performance on the query set of each of the training tasks based on the knowledge gathered from its support set. The network is given a different task at each time step, so it must learn to discriminate data classes in general rather than specific subsets.

A common way of attempting to solve a few-shot learning problem is by using prior knowledge about similarity. This is done by learning class embeddings that tend to separate classes even if they have never been seen before. One of the earlier methods for solving few-shot problems is a pairwise comparator [47,48] that was developed to classify two examples as belonging or not to the same class based on their similarity, even though the model had never seen those classes before. This method can be adapted to few-shot learning by classifying an example from the query set according to its maximum similarity to an example in the support set. A more elegant way is multi-class comparators [49,50], which learn a common representation for each class in the training set and match each new test example using cosine similarity. Snell et al.[50] propose Prototypical Networks,

Figure 1. The "Cheat-Game" interface.

which average the embeddings of the examples from the same class to compute the class prototype (mean vector). Then a distance metric is used to calculate the similarity (a negative multiple of the Euclidean distance) between each query embeddings to each of the classes' prototypes for finding the most similar class.

Alternatively, the few-shot learning problem can be solved by learning parameters that generalize better to similar tasks and can be fine-tuned very quickly when applied to different tasks. An implementation of that approach is Model-Agnostic Meta-Learning (MAML), introduced by Finn et al. [51]. The model is initialized with random weights, and iteratively, for each task in a meta-batch of tasks, it fine-tunes a copy learner using the weights of the primary model (meta-learner). The learner weights are updated using the loss from the query samples in the task by stochastic gradient descent. At the end of each training task, the losses and gradients from the queries are accumulated, the derivative of the mean loss concerning the primary model's weights is computed, and the weights in the primary model are updated. The primary model's weights improve during this process so that the model can fine-tune to other tasks faster.

Recently, the few-shot learning method was applied to the SER problem. Guibon et al. [52] use Prototypical networks for emotion sequence labeling, Feng and Chaspari [53] use Siamese Neural Network for emotion recognition of spontaneous speech and [54] use MAML for solving the Multilingual SER problem. Hence, as this approach seems promising for the SER task, we adapt it to be used in our problem.

3. Data collection

The data presented in this work is based on the "Cheat-Game", which is a turn-taking card game. Each player is dealt 8 cards and the goal is to play all cards. The centered pile accumulates all cards played by the players, and every turn the value of the recently played card is supposed to either go up by one or down by one. On each turn, a player may place up to four cards (faced down) on the centered pile. The player then states which cards she disposed; however, the player may claim to put cards that are different from what she actually played (i.e., *a false claim*). Nevertheless, the player must claim to play cards that have a value of either one above or one below the recently played card(s). If a player suspects that her opponent is cheating, i.e., played cards that are different than what she

has stated, the player may call out a cheat. If the opponent did in-fact cheat, she collects all the cards; otherwise, the player that called out a cheat collects the cards.

We use the implementation of Mansbach et al. [55] for the "Cheat-Game" (see Figure 1 for a screenshot). During game-play, the claims of the players are recorded and added to our data set along with the actual cards played. This information allows us to determine whether a claim is true or false.

To obtain high-quality results, we improved the game environment by adding an audio test at the beginning of the game and an option for the player to hear her claim so that she could make sure it sounds clear; if not, it could be re-recorded. Unfortunately, some players who played illegal cards (i.e., opted to cheat), rather than saying that they played legal cards, stated the cards they actually played. Such statements cannot be seen as an untruthful statement, nor can they be used as a truthful statement. Therefore, we attempted to identify such statements and remove them from the data set. To that end, we used the Google Speech-to-Text API, and provided to it relevant words from our domain. Then, we checked whether the players who played cards not according to the rules stated that they had played illegal cards, and if so, we have removed these samples from the data-set. That method resulted in 3350 samples being removed.

We recruited 156 test subjects who played the game in English using Amazon's Mechanical Turk service [56]. Subjects' demographic information is shown in Table 1. We collected a total of 10,788 labeled samples. 7,585 samples were labeled as true (70.3%), and 3,203 samples were labeled as false (29.7%). Each sample lasts about 4 seconds.

Table 1. Subjects' demographic information

Gender	Male	97
	Female	59
Education level	High-school	62
	Bachelor's	77
	Master's	13
	PhD	4
Average age		35.9

As mentioned, we divided the subjects into two groups: subjects whose data is used only for training and subjects used only for testing. In the training set, we had 111 subjects, and in the test set, we had 45 subjects. It should be noted that in our data set, different recordings are made in different environments, unlike most data sets in which all recordings are gathered from the same environment (i.e., the same microphones). This fact makes our problem more complex, but also more realistic, as, when used in practice, it is anticipated that the data is gathered from many different sources, and each person uses her own recording device. See Table 2 for a summary of the data-set division.

Table 2. Data-set Division

Set	**Subjects**		**Samples**
Train	Male	66	7529
	Female	45	
Test	Male	31	3259
	Female	14	

4. Comparative Hint Approach Meta-Learning (CHAML)

We present CHAML, a model that uses a Comparative Hint Approach and Meta-Learning for deception detection. The model is composed of the following three modules: the embedding module, the core process, and the classifier.

4.1. Embedding Types

In order to classify the audio samples, they must first be vectorized. Therefore, we use embeddings, which encompass the samples' audio features, and feed them to the core process. We consider two different types of embeddings: Five Sound Features and Wav2Vec 2.0.

4.1.1. Five Sound Features

The Five Sound Features is a vector embedding developed by Mansbach et al. [55]. It contains following five features, which are extracted from the audio samples: MFCC, Mel-scale spectrogram, Spectral contrast, Short-time Fourier transform (STFT), and Tonnetz. The vector embedding size is 193. We note that although [55] used Voice Activity Detector (VAD) to trim the silence parts and background noise for all samples, we observed that this practice did not improve the performance of this embedding method. This is likely because the samples are short, and silent segments may contain clues on whether a statement is true or false. Therefore, in this paper, we do not use VAD.

4.1.2. Wav2Vec 2.0

Wav2Vec 2.0 [42] is a framework for self-supervised learning of speech representations. The model training process is divided into two phases; firstly, the model is pre-trained in a self-supervised manner on large quantities of unlabeled audio samples to achieve the best speech representation it can, and secondly, it is fine-tuned on a smaller amount of labeled data.

The architecture of the model is composed of three parts: feature encoder, context network and quantization module. The feature encoder reduces the dimensionality of the input raw waveform by converting it into a sequence of T latent audio representations vectors of 25ms each. It consists of a 1-d convolutional neural network with 7 layers and 512 neurons at each layer. These representations are then fed into both the context network and the quantization module. The context network takes the T latent audio representations and processes them through Transformer blocks for adding information from the entire audio sequence and getting the contextualized representations. The quantization module discretizes the latent audio representation to a finite set of quantized representations via product quantization. The set of possible quantized representations is composed of a concatenation of codewords sampled from codebooks. The pretraining process uses contrastive learning in which it randomly masks parts of the audio from the latent feature encoder space, which requires the model to identify for each masked frame the correct quantized latent representations from a set of distractors. As mentioned, after the pre-training phase, the model is fine-tuned on labeled data.

In this study, during the fine-tuning phase, the quantization module is not used. Instead, the model is fine-tuned in a supervised task for speech classification by adding an average pooling layer on top of the context encoder output for calculating the averaged vector from the context representation, followed by a fully connected layer using the Tanh activation function and an output layer with two classes. The hyper-parameters used for fine-tuning the Wav2Vec 2.0 model on our data are represented in Table 3. We note that the total train batch size was set to the maximal allowed by the GPU used for training, and since fine-tuning the Wav2Vec 2.0 model required extensive training time, we could not consider many different hyper-parameters. This fine-tuning architecture is inspired by [57] that fine-tunes the Wav2Vec 2.0 model for the speech classification task.

The pre-trained model used in our study is the Wav2Vec 2.0 xlsr-53 model[1] that was fine-tuned on English using the Common Voice [58] data set which currently consists of 7,335 hours of transcribed speech in 60 different languages. The model outputs an embedding vector of length 1024.

[1] https://huggingface.co/jonatasgrosman/wav2vec2-large-xlsr-53-english

Table 3. Hyper-parameters used for fine-tuning the Wav2Vec 2.0 model

Parameters	
Sample frequency	16k Hz
Learning rate	1e-4
Training epochs	5
Training batch size	12
Gradient accumulation steps	3
Total train batch size	36

4.2. CHAML - Core

In this model, we were inspired by the idea of meta-learning for few-shot learning for deception detection based on speech. We define each person in our data-set as a "meta-learning task". Each task's support set contains four randomly sampled examples, two per class, and the rest are the query set. In the training phase, each sample was trained with four examples randomly sampled from the training set of the person, i.e., each sample appears as a query, and may appear as an example in the support-set of other samples. Clearly, in the evaluation phase, we do not have different pairs of labeled examples for each query sample, and we use the same four examples for all query samples of a task. As mentioned in Table. 2, there are 45 testing subjects (i.e., tasks), and for each of them, we have four labeled examples, two per class. Therefore, in total, we have 90 True and 90 False samples in the support set of the testing tasks. See sample partition details in Table 4.

Table 4. Support-Query Division

			Support Set	**Query Set**	**Total Samples**
Train	**True**	5288	ALL	ALL	7529
	False	2241			
Test	**True**	2297	90	2207	3259
	False	962	90	872	

The CHMAL model fixes the order of the provided examples by first using the true examples and then the false examples. For each class' examples, we calculated its element-wise product; this is known to have the ability to catch similarities or discrepancies between the vectors. Each pair is then concatenated with the given product and fed into three fully connected layers with the ReLU activation function and dropout. The number of neurons in each fully connected layer depends on the original embedding size, which is a different value for each of the two different embedding types. That is, for the Five Sound Features embedding mentioned in 4.1.1 the embedding size is 193, and for the Wav2Vec 2.0 embedding mentioned in 4.1.2 the embedding size is 1024. The first fully connected layer is three times the length of the embedding, which matches the size of its input (the embedding of the first example, the embedding of the second example. and the embedding of the dot product). The length of the second and final fully connected layers is twice the embedding size. The intuition behind this architecture is to allow CHAML to identify the patterns present in each class and return a general representation that contains the features that characterize each class. The results from both the positive and negative pairs are combined with the tested sample into a vector that is fed into a neural classifier. Finally, CHAML returns the probability that the tested sample belongs to each of the classes. In the training phase, the weights of the preprocess layers and the neural classifier are all updated after each epoch.

We believe that the model compares the tested sample with the examples of each class in order to find which class is more similar to the tested sample, which helps it to provide a more accurate prediction. An illustration of CHAML's architecture is depicted in Figure 2.

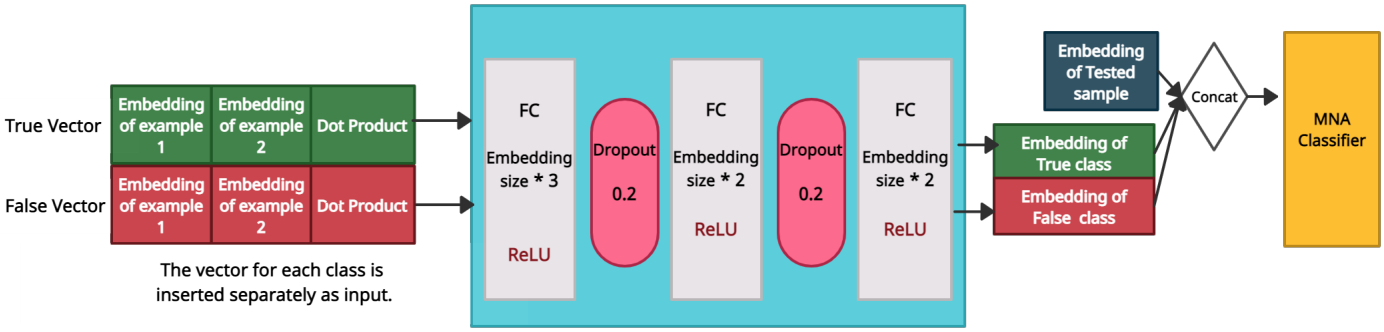

Figure 2. An Illustration of CHAML Model.

4.3. MNA Classifier

For our model's classification task, we use the same architecture as the Five Sound Feature Model (FSFM) from [55], which achieved the highest scores for the deception detection task in a very similar environment. We term this classifier MNA (Mansbach, Neiterman, and Azaria [55]). MNA's architecture consists of three fully-connected layers, using the ReLU activation function and dropout after each. The output layer uses a softmax activation function with two classes.

The classifier illustration is provided in Figure 3.

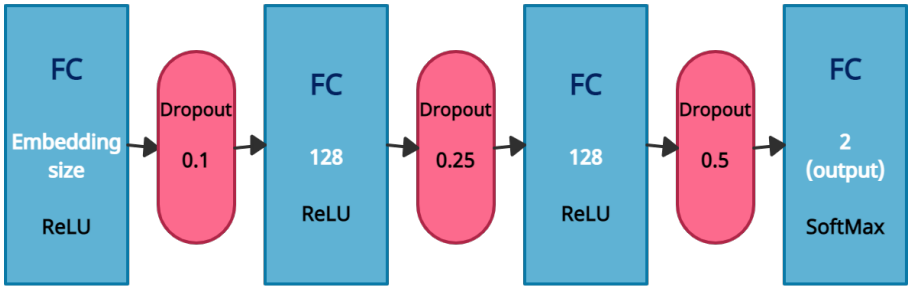

Figure 3. An Illustration of MNA Classifier.

The complete CHAML training process is presented in Algorithm 1. During the test phase, instead of randomly sampling the support set for each query sample, we use the support set provided for each task for all queries of that task.

5. Evaluation

5.1. Baseline Methods

We consider four baseline methods, and we compare their performance to that of CHAML. Namely, we consider a method using only MNA classifier without fine-tuning, and a method using MNA classifier that is fine-tuned based on the support sets. In addition, we consider the Prototypical [50] and MAML [51] networks, which are commonly used for meta-learning.

5.1.1. MNA - Fine-Tuning

In this method, we trained the MNA classifier mentioned in 4.3 on the embedding vectors. During the evaluation phase, for each test task, we first continued training the model on the four examples provided (the support set) and then predicted the value on the query of that test task.

Algorithm 1: CHAML Training Process

Input : Samples divided to tasks
Output: Predicted labels

1. Embedding Collection:

Use Five Sound Features / Wav2Vec 2.0 / other,
for creating an embedding for each of the samples.

2. CHAML-Core:

Create an empty list $model_{INPUTS}$.

a. foreach *task in train tasks* **do**
`/* create support set for each sample */`
foreach *sample in task* **do**
$T_{support}$:=Randomly sample two True samples from the current task.
$F_{support}$:=Randomly sample two False samples from the current task.
Add ($sample$, $T_{support}$, $F_{support}$) to $model_{INPUTS}$ list.
end
end

b. Shuffle $model_{INPUTS}$ list.

c. Train the model:
`/* item:` ($sample$, $T_{support}$, $F_{support}$) `*/`

for *50 epochs* **do**
foreach *item in* $model_{INPUTS}$ **do** `// batch size = 512`
foreach *class pair in support set* **do** `//` $(s1, s2)$
$pair_{MUL} := s1 * s2$ `// element-wise product`
$pair_{CON} := s1 \cdot s2 \cdot pair_{MUL}$ `// concentration`
Feed $pair_{CON}$ **into model layers:**
$FC\ (ReLU) \rightarrow (embedding\_size * 3)$
$Dropout(0.2)$
$FC\ (ReLU) \rightarrow (embedding\_size * 2)$
$Dropout(0.2)$
$FC\ (ReLU) \rightarrow (embedding\_size * 2)$
Output: $CLASS_{vec}$
`// CLASS = TRUE/FALSE`
end
3. Classifier:`// MNA`
$sample_{CON} := sample \cdot T_{vec} \cdot F_{vec}$
`// concentration`
Feed $sample_{CON}$ **into classifier layers.**
end
Update all layers parameters (including classifier layers).
end

5.1.2. Prototypical Network

A prototypical network [50] is one of the well-known meta-learning methods and is based on similarity. For each task, the model learns a prototypical embedding, which is a common representation for each class in the support set and matches each query embedding to each class's prototypical embedding to find the most similar class. The prototypical network uses cosine similarity for measuring the similarity between each query embedding and each class prototypical embedding.

5.1.3. Model-Agnostic Meta-Learning (MAML)

Model agnostic meta-learning (MAML) [51] is a meta-learning framework that learns the model parameters that can be fine-tuned very quickly when applied to different tasks. The model is initialized with random weights, and iteratively, for each task in the training tasks, fine-tunes a copy of the primary model. Then the weights of the copy are updated using the loss from the query samples in the task by stochastic gradient descent. At the end of each training epoch, the losses and gradients from all queries are accumulated. MAML then calculates the derivative of the mean loss concerning the primary model's weights, and updates those weights in the primary model. In the evaluation phase, a copy of the primary model is fine-tuned on each support set of the test tasks and evaluated on the query set of the same task.

5.2. Results

In this section, we describe our results for deception detection. We considered training the basic MNA model using 25, 50, and 100 epochs as well as batch sizes of 32, 128, 512, and 1000. Since using 50 epochs and a batch size of 512 performed best for the basic MNA model, we used these hyper-parameters for all the models that we have tested. In addition, we use weighted categorical cross-entropy loss to deal with the imbalance of our data. The results presented in this paper are the average of 30 different executions, each with a different seed. The scores are calculated on the query sets of the test tasks and can be seen in Table 5. We note that since the False samples are the minority, and those which we are trying to identify as being deceptive, we use them as the positive class, and the True samples are used as the negative class. Therefore, False samples classified as being False are considered true positives. Similarly, True samples classified as being False are considered false positives.

Table 5. Comparison of models performance

Model	Accuracy	Precision	Recall	F1-Score
MNA - *Five-features* [55]	55.82	0.2977	0.4122	0.3444
MNA - *Wav2Vec 2.0*	60.46	0.3411	0.4208	0.3745
MNA-FT - *Five-features*	55	0.3129	0.4922	0.3822
MNA-FT - *Wav2Vec 2.0*	54.3	0.3051	0.4803	0.3731
Prototypical [50] - *Five-features*	53.66	0.294	0.4547	0.3548
Prototypical [50] - *Wav2Vec 2.0*	52.24	0.2765	0.4253	0.335
MAML [51] - *Five-features*	58.8	0.3148	0.3865	0.3466
MAML [51] - *Wav2Vec 2.0*	61.28	0.349	0.424	0.3823
CHAML - *Five-features*	58.59	0.3345	0.4585	0.3826
CHAML - *Wav2Vec 2.0*	**61.34**	0.3515	0.4307	**0.3857**

As depicted by the Table, the CHAML - Wav2Vec 2.0 model outperformed all other methods with an accuracy of 61.34% and an F1-Score of 0.3857. Table 6 provides the confusion matrix for the CHAML model on the Wav2Vec 2.0 embedding.

Next, we compare CHAML to the MNA classifier. As shown in the table, CHAML outperformed the fine-tuning method and improved the results of the MNA classifier for both embedding types. We note that the FSFM original work [55] (which MNA's architecture was inspired by it) presented higher accuracy and F1-score results than in our

Table 6. CHAML - Wav2Vec 2.0 performance on the query set of the test samples

		Predicted	
		True	**False**
Actual	**True**	1513	694
	False	496	376

experiments. This is due to the fact that in the original work, the entire data set was first shuffled and then split into training and test sets. Therefore, the same person also appeared in the training set and also in the test set. This allowed the model to learn from all types of people. However, in our current work, there is no overlapping between samples of the same person in both train and test sets. Consequently, the performance decreases as the model is trained on some people but tested on others.

In addition, CHAML is compare to prototypical and MAML methods, which are commonly used for meta-learning. The Prototypical network was trained using 50 epochs for each support set and used weighted categorical cross-entropy loss. For obtaining the prototypical embedding for each class, we use the FSFM architecture without the last layer used for the classification task; the prototypical embedding length is 128. MAML was trained on 50 epochs with 15 epochs for each task's fine-tuning and used the FSFM model as the meta-learner. MAML performed better when using random weighted sampling rather than weighted categorical cross-entropy loss for accounting for the imbalance in the data. Clearly, since the Prototypical network and MAML are trained on each task separately, they must use a single support set for all the query samples, unlike CHAML, which has different support sets for each query sample in the training tasks. The results show that CHAML outperforms other meta-learning methods. The Prototypical network achieved the lowest accuracy compared to all other methods and a lower F1-score compared to CHAML, while MAML achieved similar results. The averaged computation times of the three models can be seen in table 7. As shown, CHAML is over 25 times faster than MAML while having a similar computation time to the Prototypical network, but achieves much higher performance.

Table 7. CHAML vs. Meta-Learning Methods Average Computation Times

Model	**Avg. Time in sec.**
MAML [51]	1047.71
Prototypical [50]	27.22
CHAML	41.07

Moreover, we find that using the Wav2Vec 2.0 embedding 4.1.2 for the deception detection task on our data, seems to have better results in the main models: MNA, MAML, and CHAML. Interestingly, even in the MNA classifier, which was originally specially designed for the five-sound features embedding, the Wav2Vec 2.0 embedding performs better.

In order to confirm that CHAML uses the examples and does not ignore them, we conduct another experiment in which the support set was mixed and did not have the two true examples first and the two false examples later (as required by CHAML). In this case, the F1-score has significantly decreased from 0.3857 to 0.3707 in the Wav2Vec 2.0 setting and from 0.3826 to 0.332 in the Five-Features setting. This indicates that CHAML relies on the examples for providing an accurate prediction.

6. Conclusions & Future work

In this study, we have proposed CHAML, a comparative hint approach meta-learning for deception detection based on speech. The method is based on the meta-learning approach in which a model is trained on various learning tasks to enable it to solve new learning tasks using only a few samples. In our approach, we added some labeled samples (the support set) to each unlabeled sample (the query set) from the same task to predict its label. Our setting differs from the typical meta-learning problem since our data comes from different environments (and different people), whereas in the classical meta-learning framework, data from one environment is divided into different tasks. In addition, in typical meta-learning, there is no overlap between the classes in the training tasks and those in the test task. However, in our approach, classes are the same in all tasks, but the task environment differentiates each one from the other. Therefore, typical meta-learning methods do not perform well in our task, while CHAML, our proposed method, manages to gather relevant information from the support sets and improves the model's performance.

In future work we attempt to find a way to add attention to CHAML, so that it learns how to attend to the relevant parts of each example when making a prediction. We note that our method is not limited to deception detection and can also be applied to other environments, in which all tasks, both in training set and in testing set, include the same classes (or regression problems), but the samples in different tasks highly differ from each other. Examples of such environments include emotion recognition (with many different people; each person is a different task), handwriting recognition (with different people; each person is a different task), and age and sex determination of different animals (each animal is its task), and determining whether sensor data collected from different environments indicates that human intervention is required. In future work, we intend to test CHAML on some of these environments.

Author Contributions: Conceptualization, N.M. and A.A.; methodology, N.M. and A.A; software, N.M.; validation, N.M. and A.A.; formal analysis, N.M. and A.A..; investigation, A.A.; data curation, N.M.; writing—original draft preparation, N.M.; writing—review and editing, N.M. and A.A.; visualization, N.M.; supervision, A.A.; project administration, A.A.; funding acquisition, A.A. Both authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by the Ministry of Science, Technology & Space, Israel.

Institutional Review Board Statement: The study was conducted in accordance with the Declaration of Helsinki, and approved by the Institutional Review Board (or Ethics Committee) of Ariel University (AU-NAT-AA-20210816 Aug 16, 2021).

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: All data is available at: deception_data.zip

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CHAML	Comparative Hint Approach Meta Learning
SER	Speech Emotion Recognition
MAML	Model Agnostic Meta Learning
FSFM	Five Sound Features Model
MNA	Classifier from Mansbach, Neiterman & Azaria [55]

References

1. Ekman, P.; Friesen, W.V. *Unmasking the face: A guide to recognizing emotions from facial clues*; Ishk, 2003.
2. Dechêne, A.; Stahl, C.; Hansen, J.; Wänke, M. The truth about the truth: A meta-analytic review of the truth effect. *Personality and Social Psychology Review* **2010**, *14*, 238–257.
3. Trovillo, P.V. History of lie detection. *Am. Inst. Crim. L. & Criminology* **1938**, *29*, 848.

4. Ott, M.; Choi, Y.; Cardie, C.; Hancock, J.T. Finding deceptive opinion spam by any stretch of the imagination. *arXiv preprint arXiv:1107.4557* **2011**.
5. Herchonvicz, A.L.; Santiago, R.d. Deep Neural Network Architectures for Speech Deception Detection: A Brief Survey. In Proceedings of the EPIA Conference on Artificial Intelligence. Springer, 2021, pp. 301–312.
6. Pérez-Rosas, V.; Abouelenien, M.; Mihalcea, R.; Burzo, M. Deception detection using real-life trial data. In Proceedings of the Proceedings of the 2015 ACM on International Conference on Multimodal Interaction, 2015, pp. 59–66.
7. Council, N.R.; et al. *The polygraph and lie detection*; National Academies Press, 2003.
8. Podlesny, J.A.; Raskin, D.C. Physiological measures and the detection of deception. *Psychological bulletin* **1977**, *84*, 782.
9. Thackray, R.I.; Orne, M.T. A COMPARISON OF PHYSIOLOGICAL INDICES IN DETECTION OE DECEPTION. *Psychophysiology* **1968**, *4*, 329–339.
10. Vrij, A. *Detecting lies and deceit: The psychology of lying and implications for professional practice*; Wiley, 2000.
11. Knapp, M.L.; Hart, R.P.; Dennis, H.S. An Exploration of Deception as a Communication Construct. *Human Communication Research* **2006**, *1*, 15–29.
12. Dulaney Jr, E.F. Changes in language behavior as a function of veracity. *Human Communication Research* **1982**, *9*, 75–82.
13. DePaulo, B.M.; Lindsay, J.J.; Malone, B.E.; Muhlenbruck, L.; Charlton, K.; Cooper, H. Cues to deception. *Psychological bulletin* **2003**, *129*, 74.
14. Aamodt, M.G.; Custer, H. Who can best catch a liar? *Forensic Examiner* **2006**, *15*, 6.
15. Feng, S.; Banerjee, R.; Choi, Y. Syntactic stylometry for deception detection. In Proceedings of the Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), 2012, pp. 171–175.
16. Barsever, D.; Singh, S.; Neftci, E. Building a better lie detector with BERT: The difference between truth and lies. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN). IEEE, 2020, pp. 1–7.
17. Hirschberg, J.B.; Benus, S.; Brenier, J.M.; Enos, F.; Friedman, S.; Gilman, S.; Girand, C.; Graciarena, M.; Kathol, A.; Michaelis, L.; et al. Distinguishing deceptive from non-deceptive speech **2005**.
18. Nasri, H.; Ouarda, W.; Alimi, A. ReLiDSS: Novel lie detection system from speech signal. *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)* **2016**, pp. 1–8.
19. Muda, L.; Begam, M.; Elamvazuthi, I. Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW) techniques. *arXiv preprint arXiv:1003.4083* **2010**.
20. Ittichaichareon, C.; Suksri, S.; Yingthawornsuk, T. Speech recognition using MFCC. In Proceedings of the International conference on computer graphics, simulation and modeling, 2012, Vol. 9.
21. Lalitha, S.; Geyasruti, D.; Narayanan, R.; Shravani, M. Emotion detection using MFCC and cepstrum features. *Procedia Computer Science* **2015**, *70*, 29–35.
22. Lee, C.H.; Chou, C.H.; Han, C.C.; Huang, R.Z. Automatic recognition of animal vocalizations using averaged MFCC and linear discriminant analysis. *pattern recognition letters* **2006**, *27*, 93–101.
23. Sitaula, C.; He, J.; Priyadarshi, A.; Tracy, M.; Kavehei, O.; Hinder, M.; Withana, A.; McEwan, A.; Marzbanrad, F. Neonatal bowel sound detection using convolutional neural network and Laplace hidden semi-Markov model. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* **2022**.
24. Burne, L.; Sitaula, C.; Priyadarshi, A.; Tracy, M.; Kavehei, O.; Hinder, M.; Withana, A.; McEwan, A.; Marzbanrad, F. Ensemble Approach on Deep and Handcrafted Features for Neonatal Bowel Sound Detection. *IEEE Journal of Biomedical and Health Informatics* **2022**.
25. Graciarena, M.; Shriberg, E.; Stolcke, A.; Enos, F.; Hirschberg, J.; Kajarekar, S. Combining Prosodic Lexical and Cepstral Systems for Deceptive Speech Detection. In Proceedings of the 2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings, 2006, Vol. 1, pp. I–I. https://doi.org/10.1109/ICASSP.2006.1660200.
26. Marcolla, F.M.; de Santiago, R.; Dazzi, R.L. Novel Lie Speech Classification by using Voice Stress. In Proceedings of the ICAART (2), 2020, pp. 742–749.
27. Xie, Y.; Liang, R.; Tao, H.; Zhu, Y.; Zhao, L. Convolutional Bidirectional Long Short-Term Memory for Deception Detection With Acoustic Features. *IEEE Access* **2018**, *6*, 76527–76534. https://doi.org/10.1109/ACCESS.2018.2882917.
28. Ding, M.; Zhao, A.; Lu, Z.; Xiang, T.; Wen, J.R. Face-focused cross-stream network for deception detection in videos. In Proceedings of the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 7802–7811.
29. Stathopoulos, A.; Han, L.; Dunbar, N.; Burgoon, J.K.; Metaxas, D. Deception Detection in Videos Using Robust Facial Features. In Proceedings of the Proceedings of the Future Technologies Conference. Springer, 2020, pp. 668–682.
30. Mathur, L.; Matarić, M.J. Introducing representations of facial affect in automated multimodal deception detection. In Proceedings of the Proceedings of the 2020 International Conference on Multimodal Interaction, 2020, pp. 305–314.
31. Monaro, M.; Maldera, S.; Scarpazza, C.; Sartori, G.; Navarin, N. Detecting deception through facial expressions in a dataset of videotaped interviews: A comparison between human judges and machine learning models. *Computers in Human Behavior* **2022**, *127*, 107063.
32. Wu, Z.; Singh, B.; Davis, L.; Subrahmanian, V. Deception detection in videos. In Proceedings of the Proceedings of the AAAI conference on artificial intelligence, 2018, Vol. 32.
33. Gogate, M.; Adeel, A.; Hussain, A. Deep learning driven multimodal fusion for automated deception detection. In Proceedings of the 2017 IEEE symposium series on computational intelligence (SSCI). IEEE, 2017, pp. 1–6.

34. Krishnamurthy, G.; Majumder, N.; Poria, S.; Cambria, E. A deep learning approach for multimodal deception detection. *arXiv preprint arXiv:1803.00344* **2018**.
35. Nwe, T.; Foo, S.; Silva, L.D. Speech emotion recognition using hidden Markov models. *Speech Communication* **2003**, *41*, 603–623.
36. Jain, M.; Narayan, S.; Bhowmick, P.B.A.; Muthu, R.; Bharath, K.; Karthik, R. Speech emotion recognition using support vector machine.
37. F., N.; T., S.; D., K.; G., A. Vocal-based emotion recognition using random forests and decision tree. *Int. J. Speech Technol.* **2017**, *25*, 1–8.
38. Khalil, R.A.; Jones, E.; Babar, M.I.; Jan, T.; Zafar, M.H.; Alhussain, T. Speech emotion recognition using deep learning techniques: A review. *IEEE Access* **2019**, *7*, 117327–117345.
39. Trigeorgis, G.; Ringeval, F.; Brueckner, R.; Marchi, E.; Nicolaou, M.A.; Schuller, B.; Zafeiriou, S. Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network. In Proceedings of the 2016 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, 2016, pp. 5200–5204.
40. Han, K.; Yu, D.; Tashev, I. Speech emotion recognition using deep neural network and extreme learning machine. In Proceedings of the Proceedings of INTERSPEECH ISCA Singapore, 2014.
41. Fayek, H.M.; Lech, M.; Cavedon, L. Evaluating deep learning architectures for speech emotion recognition. *Neural Networks* **2017**, *92*, 60–68.
42. Alexei, B.; Henry, Z.; Abdelrahman, M.; Michael, A. wav2vec 2.0: A framework for self-supervised learning of speech representations. *arXiv preprint arXiv:2006.11477* **2020**.
43. Pepino, L.; Riera, P.; Ferrer, L. Emotion recognition from speech using wav2vec 2.0 embeddings. *arXiv preprint arXiv:2104.03502* **2021**.
44. Chen, L.W.; Rudnicky, A. Exploring Wav2vec 2.0 fine-tuning for improved speech emotion recognition. *arXiv preprint arXiv:2110.06309* **2021**.
45. Vaessen, N.; Van Leeuwen, D.A. Fine-Tuning Wav2Vec2 for Speaker Recognition. In Proceedings of the ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2022, pp. 7967–7971. https://doi.org/10.1109/ICASSP43922.2022.9746952.
46. Thrun, S.; Pratt, L. Learning to learn: Introduction and overview. In *Learning to learn*; Springer, 1998; pp. 3–17.
47. Koch, G.; Zemel, R.; Salakhutdinov, R.; et al. Siamese neural networks for one-shot image recognition. In Proceedings of the ICML deep learning workshop. Lille, 2015, Vol. 2, p. 0.
48. Hoffer, E.; Ailon, N. Deep metric learning using triplet network. In Proceedings of the International workshop on similarity-based pattern recognition. Springer, 2015, pp. 84–92.
49. Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D.; et al. Matching networks for one shot learning. *Advances in neural information processing systems* **2016**, *29*.
50. Snell, J.; Swersky, K.; Zemel, R. Prototypical networks for few-shot learning. *Advances in neural information processing systems* **2017**, *30*.
51. Finn, C.; Abbeel, P.; Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In Proceedings of the International conference on machine learning. PMLR, 2017, pp. 1126–1135.
52. Guibon, G.; Labeau, M.; Flamein, H.; Lefeuvre, L.; Clavel, C. Few-shot emotion recognition in conversation with sequential prototypical networks. *arXiv preprint arXiv:2109.09366* **2021**.
53. Feng, K.; Chaspari, T. Few-shot learning in emotion recognition of spontaneous speech using a siamese neural network with adaptive sample pair formation. *IEEE Transactions on Affective Computing* **2021**.
54. Naman, A.; Sinha, C. Fixed-MAML for Few-shot Classification in Multilingual Speech Emotion Recognition. In *Machine Intelligence and Smart Systems*; Springer, 2022; pp. 473–483.
55. Mansbach, N.; Neiterman, E.H.; Azaria, A. An Agent for Competing with Humans in a Deceptive Game Based on Vocal Cues. *Proc. Interspeech 2021* **2021**, pp. 4134–4138.
56. Paolacci, G.; Chandler, J.; Ipeirotis, P.G. Running experiments on amazon mechanical turk. *Judgment and Decision making* **2010**, *5*, 411–419.
57. Pascal, F.; Dominique, R. Speech Classification using wav2vec 2.0, 2021. https://www.zhaw.ch/storage/engineering/institute-zentren/cai/BA21_Speech_Classification_Reiser_Fivian.pdf.
58. Rosana, A.; Megan, B.; Kelly, D.; Michael, H.; Kohler, M.; Meyer, J.; Morais, R.; Saunders, L.; Tyers, F.M.; Weber, G. Common voice: A massively-multilingual speech corpus. *arXiv preprint arXiv:1912.06670* **2019**.