Classification with Meta-Learning

- EMMANOUIL KOUTOULAKIS
- EMMANOUIL MARKODIMITRAKIS
- YAĞMUR ÇIĞDEM AKTAŞ

Workflow

Research ✓	SOTA meta-learning papers with high citations
Explore ✓	meta-learning approaches
Select ✓	a paper based on the citations
Insights ✓	Study the paper and find the most important details
Implement	the Meta-Learning model using a Deep Learning framework (Tensorflow or PyTorch)
Evaluate ✗	Evaluate the model, using proposed metrics
Compare ✗	Compare the proposed method with other SOTA methods

Explore meta-learning approaches

There are three common approaches[8]

- Using (cyclic) networks with external or internal memory (model-based)
- **Learning effective distance metrics (metrics-based)** ✓
- Explicitly optimizing model parameters for fast learning (optimization-based).

Research SOTA Meta-Learning Papers

We selected the most recent approaches about Meta Learning in the following citation databases

Search criteria: ***"Meta learning" AND "Covid" AND "Diagnosis"***

We ordered the outputs of databases according to the citation score and the publication date

Literature Data-bases	**Search Results**
Science Direct	4
Google Scholar	358
PubMed	3
Sum	365

Select a paper based on the citations

Publication	Citations	Title
[1] M. Shorfuzzaman, M. S. Hossain	40	MetaCOVID: A Siamese neural network framework with contrastive loss for n-shot diagnosis of COVID-19 patients
[2] T. Naren, Y. Zhu et al.	0	COVID-19 diagnosis using model agnostic meta-learning on limited chest X-ray images
[3] R. Singh, V. Bharti, et al.	0	MetaMed: Few-shot medical image classification using gradient-based meta-learning
[4] W. Zheng, L Yan, et al.	2	Learning to learn by yourself: Unsupervised meta-learning with self-knowledge distillation for COVID-19 diagnosis from pneumonia cases

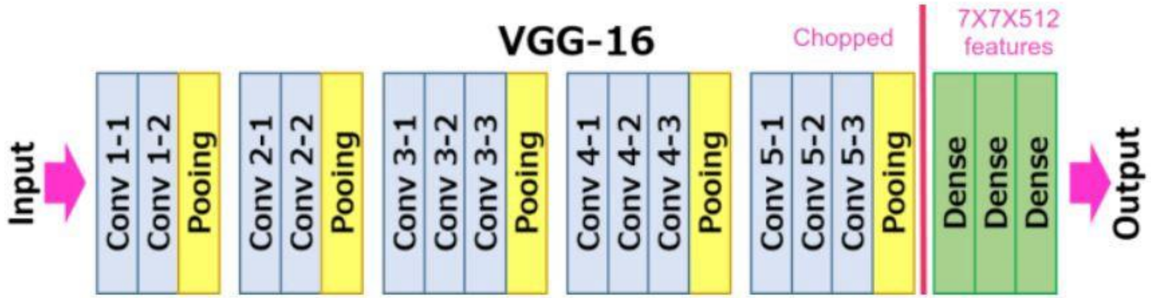
Insights

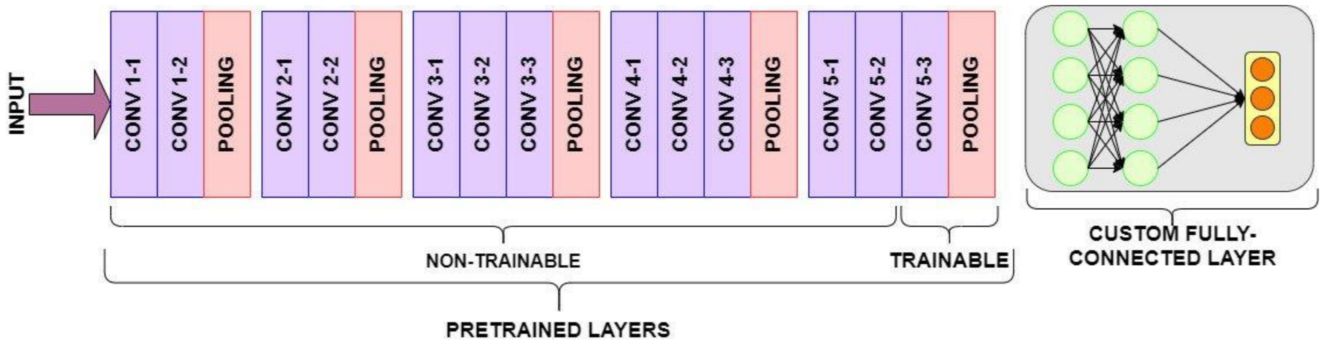
- Datasets
- Siamese Neural Network Architecture
- Loss Functions
- Training Strategy & Optimizer
- Evaluation metrics

Implement the Meta-Learning model

- Transfer learning - VGG

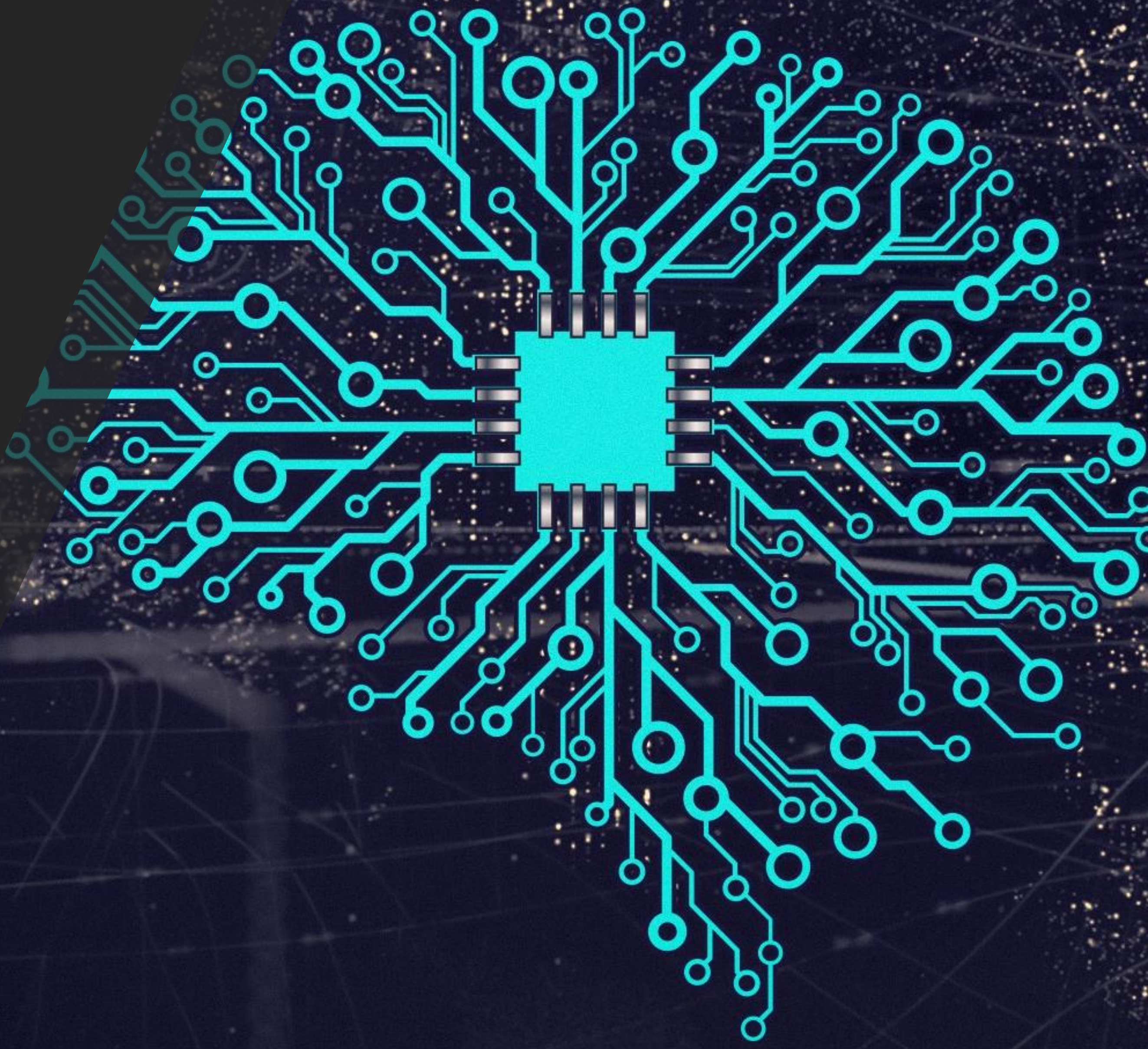
Transfer learning - VGG



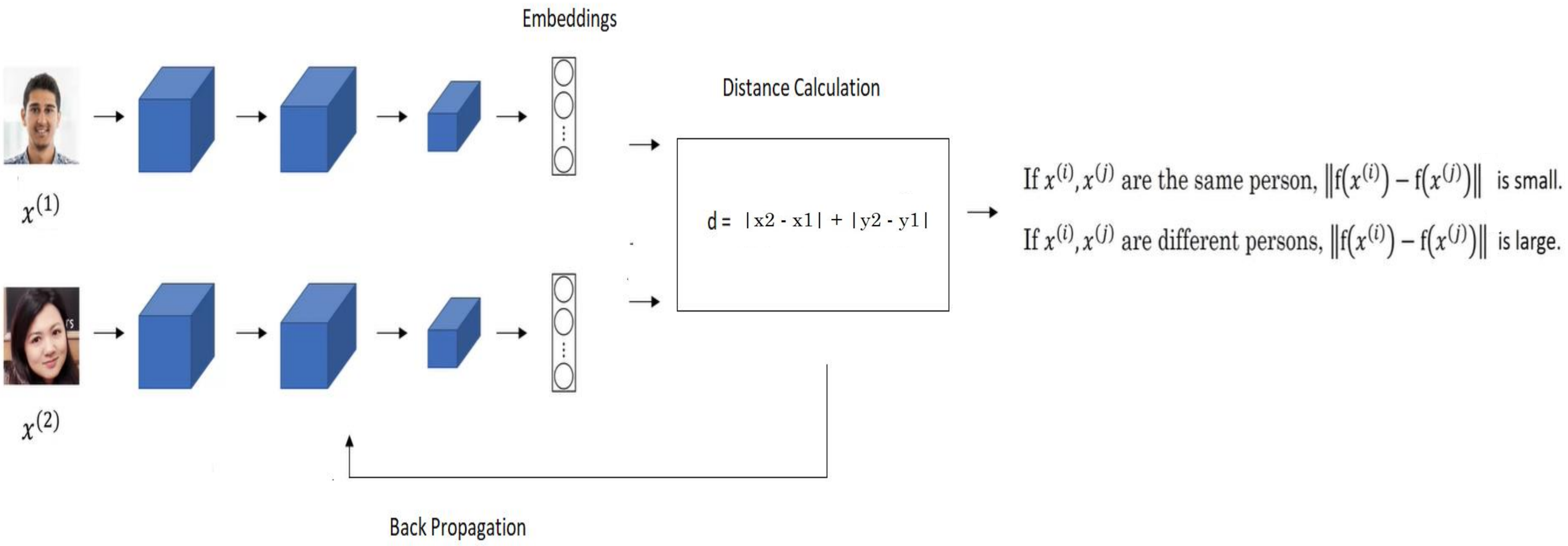


MetaCovid - A Siamese neural network framework with contrastive loss for *n* -shot diagnosis of COVID-19 patients

1. The Goal of the Paper
2. Dataset
3. Siamese Neural Network Architecture
4. Loss Function and Training Strategy
5. Experiments & Results



What is Siamese network

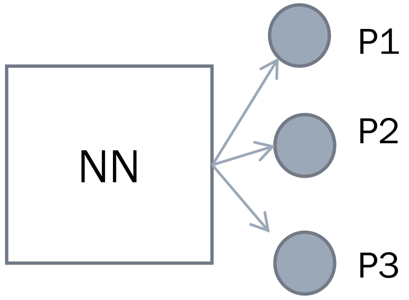


Why is it useful?

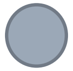
* No need for big amount of samples for a class while training

* It doesn't require to retrain the model or any change in the neural network architecture when class number is changed.

Face Recognition model for 3 person in the company



A fourth person get hired



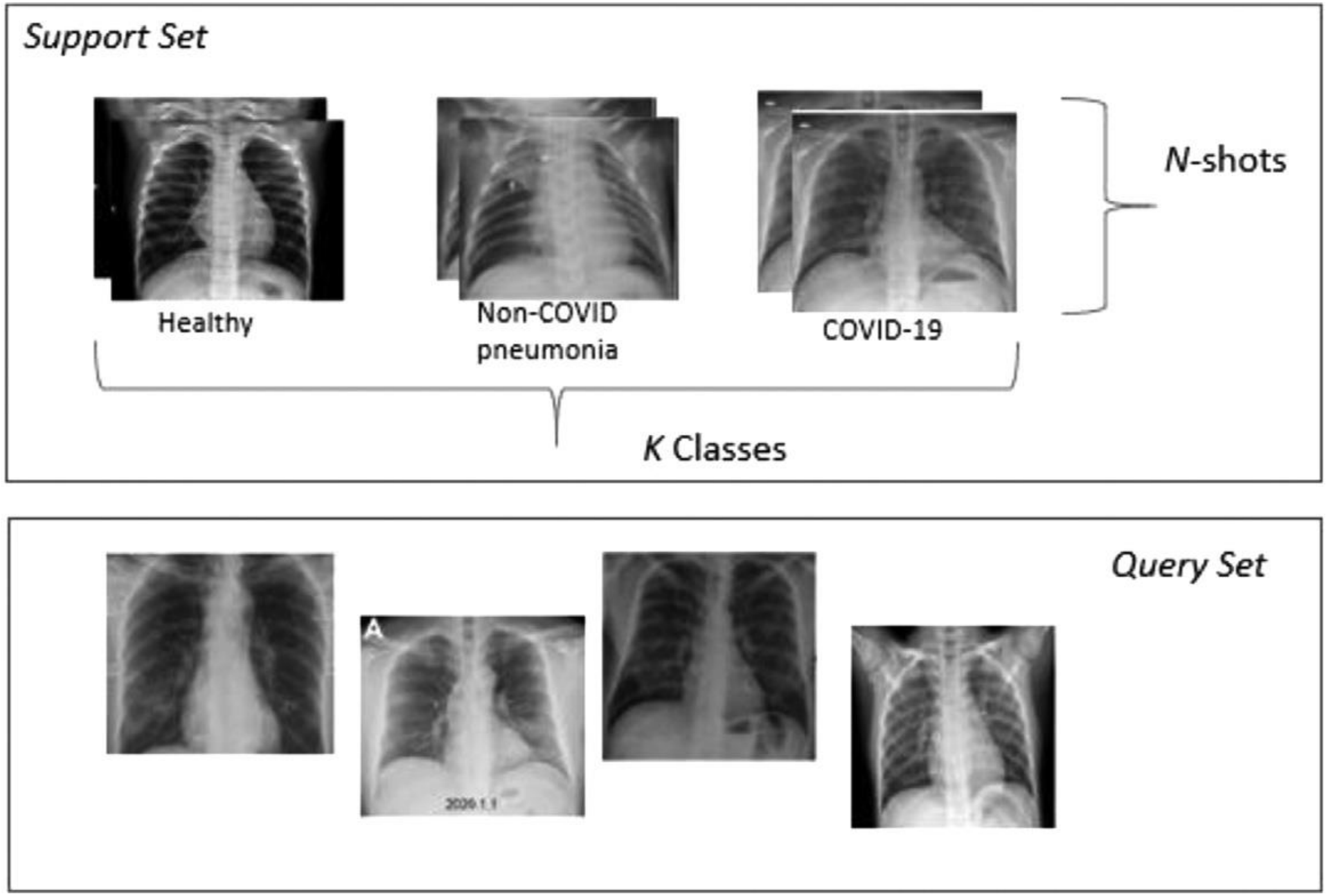
Model doesn't work anymore!!

The Goal of the Paper

- CoronaVirus detection and the methods used for this goal
- The disadvantages of the other related works
- Presentation of the method that this paper uses for COVID detection

Dataset

CXR (chest x-ray) images from open source database created by Dr. Joseph Cohen which includes COVID-19 examples and a Kaggle repository of CXR images of pneumonia and healthy patients



Classes:

- Healthy
- Non-Covid Phenomenia
- COVID images

Class	Pre-training of VGG-16 encoder network		Siamese network (n-shot learning)	
	Training	Testing	Training	Testing
Normal	160	66	10	216
Non-COVID pneumonia	160	66	10	216
COVID-19	160	66	10	216
Total	480	198	30	648

Siamese Neural Network Architecture for MetaCovid

- Pre-trained vs Custom CNNs
- Architecture
- Energy function

Pre-trained CNN vs Custom CNN

Custom Network

Small

Computationally efficient

Requires large training data to produce rich feature encoding

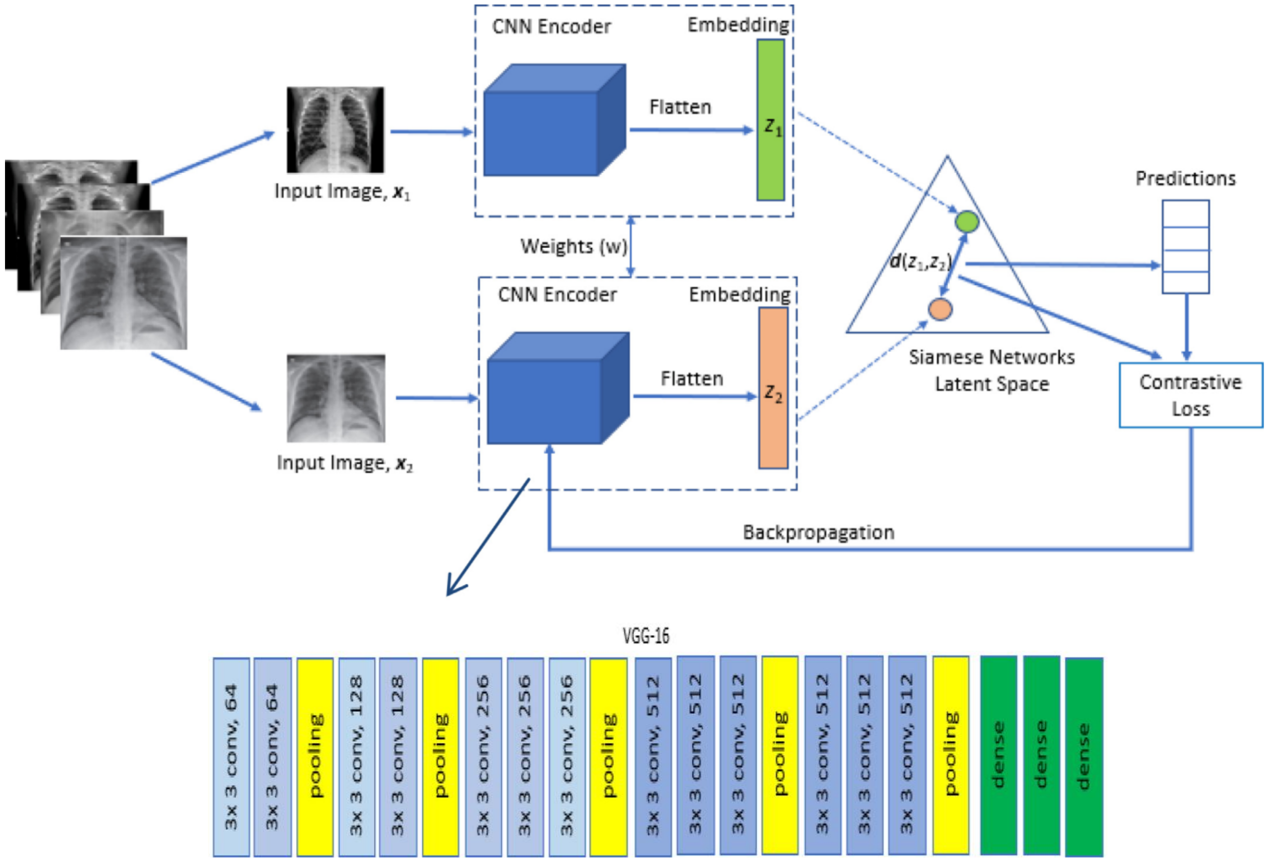
Pre-trained Network

Less training time

Requires small training data

"Hence, we have used a fine-tuned pre-trained VGG-16 [1] on large ImageNet [2] data as base encoder to obtain feature embeddings from the input images to ultimately compute similarity among them."

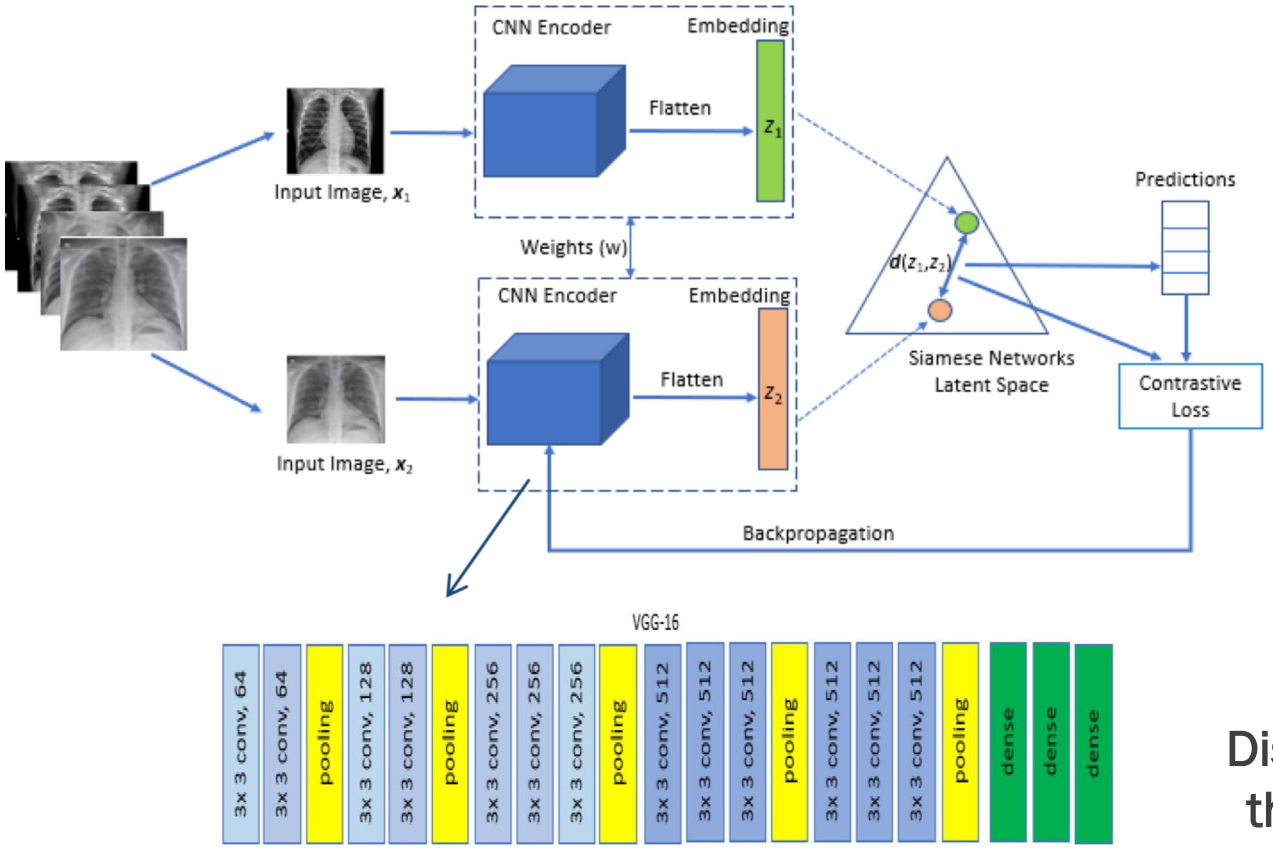
Architecture



- Two identical parallel VGG-16 networks
- Sharing the same weights and architecture
- Different input images
- Combine the output to make the final prediction

Learn a function to produce the similarity output between these two images

Energy Function



Feature embedding of x_1 denoted as $z_1(x_1)$, this is the output generated from the average pooling layer.

We get a different feature embedding $z_2(x_2)$ for the second input image.

$$E_W(x_1, x_2) = d_W(x_1, x_2) = \|z_1(x_1, x_2) - z_2(x_1, x_2)\|$$

Energy function will give us the **similarity** between the two inputs.

Distance value can be incorporated in loss function to tune the base encoder through back propagation for improved feature embeddings.

Loss Function and Training Strategy

- Contrastive Loss function
- Binary cross entropy
- Training Algorithm

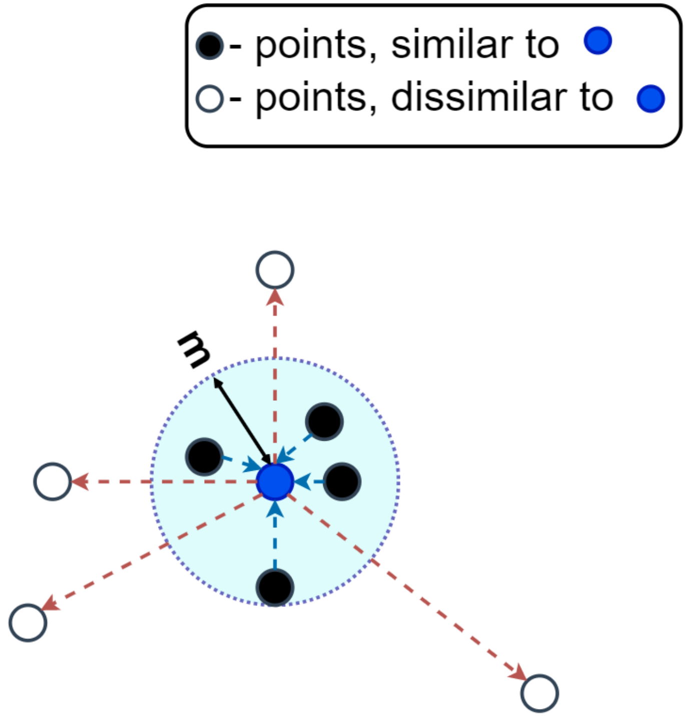
Binary cross entropy

Binary cross entropy compares each of the predicted probabilities to actual class output which can be either 0 or 1.

- The loss value will increase if the predicted probability deviates from the true label
- $L = -y\log(p) + (1-y)\log(1-p)$
  - Y is the class label (0 similar, 1 dissimilar)
  - p is the prediction probability
  - The output probability is ranging from 0 to 1.
- The equation above is used to train the network so that differentiate between similar and dissimilar images if we provide one training example from positive and negative categories and aggregate
- $L = L_{pos} + L_{neg}$

Contrastive Loss function [3]

- The loss function makes the model produce more similar feature embeddings if the target classes are the same and vice versa.
- $Loss = (1 - y) * \frac{1}{2}(d_w)^2 + (y) * \frac{1}{2}\{\max(0, m - (d_w)\}\}^2$
  - y is the true label (0 if the inputs are deemed similar and 1 are not similar)
  - d_w is the distance between the feature embeddings of the input image
- If y=1 then, the loss function will be simplified and the d_w will be maximized to m (hyper parameter margin), hence, they do not incur a loss



Training Algorithm

Algorithm 1
Training algorithm for k-way n-shot learning.

Input: Batch size N, Number of epochs *numEpochs*, Dataset D, fine-tuned VGG-16 encoder model M with parameter Θ, Loss function L, margin m
Initialize *posPairs*, *negPairs*, *posDist*, *negDist* for training and validation
D_{train}, D_{test} = split dataset, D
$\Theta_0 = w_0$
for i **do** *numEpochs*
  for b **do** getBatches()
    X_b, Y_b = random batch from D_{train}
    $posPairs = getPositivePairs(X_b, Y_b)$
    $negPairs = getNegativePairs(X_b, Y_b)$
    $posDist_b = L1\_distance(M\ (posPairs, \Theta_b))$ using Eq. (2)
    $negDist_b = L1\_distance(M\ (negPairs, \Theta_b))$ using Eq. (2)
    $dist_b = \text{concat}(posDist_b, negDist_b)$
    $L_b = Loss\ (dist_b, m, posPairs, negPairs)$ using Eq. (4)
    Update parameter Θ_b with new *weight*, w
  end for
end for

Experiments & Results

- Pre-processing
- Experimental settings
- Results
- Evaluation & Discussion

Pre-processing

- Re-scale all images to a size of 100 × 100 pixels
- Intensity normalization
- Image pixel values from [0, 255] to [0, 1]
- Histogram equalization on the input images in all three RGB channels to improve image contrast

Experimental settings

Towards the end of the pretrained model we add:

- A **flatten layer**
- Followed by a **dense layer with 5120 neurons, sigmoid** activation function, and **L2 kernel regularizer** (with a large number of kernels)
- Encodings (feature vectors) of the two input images are generated using this preceding dense layer.
- Then, we add a **customized layer to compute L1 distance** by taking the absolute difference between the encodings
- Finally, we add a **dense layer with a sigmoid** unit to generate the similarity score. We have used both contrastive and binary cross-entropy loss functions for model learning
- In addition, **Adam optimizer** is used for model training and optimizing with an **initial learning rate of 0.0001**

Results (1/3) - Contrastive vs Cross-entropy loss

Table 2
Performance results for various *n*-shot settings with contrastive loss. 3-way represents 3-class labels.

Model	Accuracy	Precision	Recall	Specificity	F1-score	AUC
MetaCOVID (3-way, 7-shot)	0.925	0.945	0.936	0.953	0.940	0.955
MetaCOVID (3-way, 8-shot)	0.936	0.951	0.945	0.965	0.938	0.962
MetaCOVID (3-way, 9-shot)	0.948	0.966	0.955	0.975	0.947	0.974
MetaCOVID 3-way, 10-shot)	0.956	0.970	0.960	0.980	0.965	0.975

Table 3
Performance results for various 3-way, *n*-shot settings with cross entropy loss.

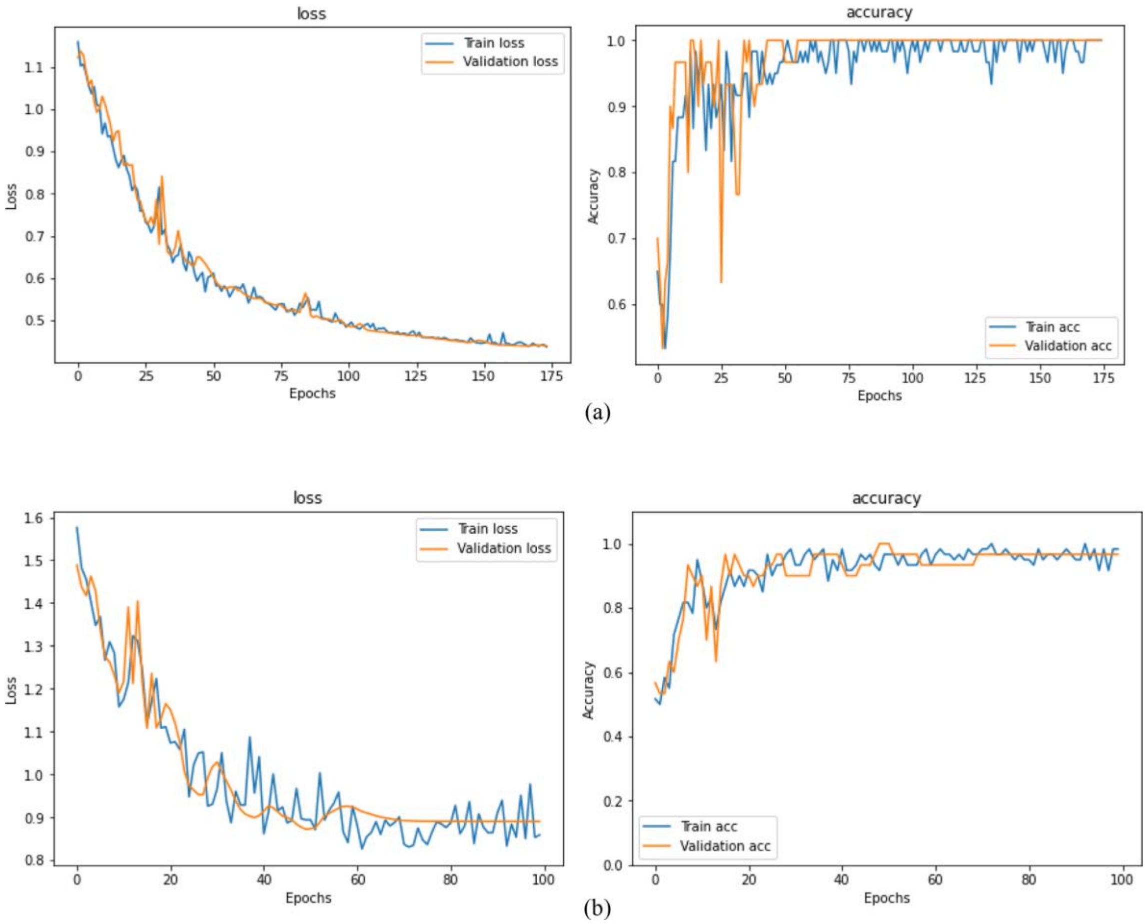
Model	Accuracy	Precision	Recall	Specificity	F1-score	AUC
MetaCOVID (3-way, 7-shot)	0.890	0.927	0.915	0.935	0.916	0.933
MetaCOVID (3-way, 8-shot)	0.915	0.935	0.919	0.940	0.922	0.948
MetaCOVID (3-way, 9-shot)	0.923	0.938	0.939	0.948	0.938	0.954
MetaCOVID 3-way, 10-shot)	0.938	0.949	0.953	0.964	0.950	0.957

Generally, the performance results obtained with **contrastive loss** function seem to be **better** than the results obtained with cross-entropy loss function.

Results (2/3) - Accuracy and loss for 3-way, 10-shot

Contrastive

Cross-entropy



The model training and validation with **contrastive loss** function appears to be **more stable** and further shows **better convergence** even though with longer training epochs

Results (3/3)

Table 4
Performance comparison between the proposed Siamese network model (with 3-way, 10-shot learning) and other pre-trained CNN models.

Model	Acc.	Precision	Recall	Specificity	F1-score	AUC
InceptionV3	0.875	0.826	0.950	0.800	0.883	0.900
Xception	0.955	0.977	0.956	0.988	0.966	0.980
InceptionResNetV2	0.900	0.833	1.00	0.800	0.908	0.900
VGG-16	0.933	0.956	0.956	0.976	0.956	0.954
MetaCOVID (3-way, 10-shot)	0.956	0.970	0.960	0.980	0.965	0.975

Table 5
Performance results of our model with contrastive loss for various 2-way, *n*-shot settings for 2-class (normal, COVID-19) classification.

Model	Accuracy	Precision	Recall	Specificity	F1-score	AUC
MetaCOVID (2-way, 7-shot)	0.940	0.955	0.945	0.958	0.949	0.965
MetaCOVID (2-way, 8-shot)	0.948	0.963	0.955	0.975	0.958	0.975
MetaCOVID (2-way, 9-shot)	0.950	0.975	0.965	0.980	0.969	0.982
MetaCOVID 2-way, 10-shot)	0.965	0.980	0.970	0.984	0.974	0.989

The proposed model produces impressive values of sensitivity (96.0%) and specificity (98.0%) which are deemed to be very critical performance estimates for applications in medical settings

Conclusion & Future Works

The proposed model exhibits comparable or in some cases better performance than the studied fine-tuned pre-trained CNN models.

It is planned to better tackle COVID-19 diagnosis problem as a multi-modal data fusion problem where various types of clinical data such as patient vitals, location, and population density will be used in addition to image data.

Resources

1. Shorfuzzaman, M., & Hossain, M. S. (2021). MetaCOVID: A Siamese neural network framework with contrastive loss for n-shot diagnosis of COVID-19 patients. *Pattern Recognition*, *113*, 107700. https://doi.org/10.1016/J.PATCOG.2020.107700

2. Naren, T., Zhu, Y., & Wang, M. D. (2021). Covid-19 diagnosis using model agnostic meta-learning on limited chest X-ray images. *Proceedings of the 12th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics*. https://doi.org/10.1145/3459930.3469517

3. Singh, R., Bharti, V., Purohit, V., Kumar, A., Singh, A. K., & Singh, S. K. (2021). MetaMed: Few-shot medical image classification using gradient-based meta-learning. *Pattern Recognition*, *120*, 108111. https://doi.org/10.1016/j.patcog.2021.108111

4. Zheng, W., Yan, L., Gou, C., Zhang, Z., Zhang, J. J., Hu, M., & Wang, F. (2021). Learning to learn by yourself: Unsupervised meta-learning with self-knowledge distillation for COVID-19 diagnosis from pneumonia cases. *International Journal of Intelligent Systems*, *36*(8), 4033-4064. https://doi.org/10.1002/int.22449

5. Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. https://arxiv.org/abs/1409.1556v6

6. Deng, J., Dong, W., Socher, R., Li, L.-J., Kai Li, & Li Fei-Fei. (2010). *ImageNet: A large-scale hierarchical image database*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

7. Hadsell, R., Chopra, S., & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2*, 1735–1742. https://doi.org/10.1109/CVPR.2006.100