Capsule Networks and Face Recognition

Adele Chui
Systems Design Engineering
University of Waterloo
Waterloo, Canada
aschui@uwaterloo.ca

Anshuman Patnaik
Systems Design Engineering
University of Waterloo
Waterloo, Canada
a3patnaik@uwaterloo.ca

Krishn Ramesh
Systems Design Engineering
University of Waterloo
Waterloo, Canada
kramesh@uwaterloo.ca

Linda Wang
Systems Design Engineering
University of Waterloo
Waterloo, Canada
ly8wang@uwaterloo.ca

***Abstract*—Although CNNs are widely used in vision tasks and achieve high accuracies, they can still fail on unseen data as they handle images quite differently than the brain. CNNs do not encode position and orientation relationships between features and so can be easily tricked. Capsule Networks were conceived by Hinton et al. as a more robust architecture capable of storing pose information and spatial relationships to recognize objects more like the brain does. Lower-level capsules store 8 dimensional vectors of features such as position, hue, texture etc. which they route to higher-level capsules using a novel routing by agreement algorithm. This gives capsule networks viewpoint invariance–which has so far eluded CNNs.**

Capsule Networks have produced great results on the MNIST dataset [3], and this paper extends the previous work to apply Capsule Networks to a face recognition task on the Labeled Faces in the Wild dataset. The trained model achieves a test accuracy of 93.7% and performs well on unseen faces that were rotated or blurred, matching or beating the performance of state-of-the-art CNNs. From previous results and the results of this paper, it can be concluded that capsule networks can outperform deeper CNNs on unseen transformed data due to their unique equivariance properties.

***Keywords*—Capsule Network, Face Recognition, LFW Dataset**

I. Problem & Motivation

Convolutional networks (CNNs) are currently the state-of-the-art when it comes to any computer vision related task, used widely in everything from object recognition systems to helping autonomous cars see. However, they fall short in three distinct and important ways that emphasize how far removed they are from how the brain processes vision [1] which can be easily seen when dealing with the problem of facial recognition. First, there is no encoding of an object's orientation and position, a major issue for early CNNs. While this was somewhat solved by the use of augmented images, CNNs still fail to recognize a transformed object if its position and orientation are not included in its training data. It would require an infinite amount of data to generalize to all viewpoints. Secondly, the mechanism that CNNs use to make predictions is easily tricked. A CNN looks at an image and extracts meaningful features that are then used to classify it accordingly. However, if all components are present but in different locations, the CNN will still classify it as the object despite the jumbled feature placement. This is typically caused by the pooling layer performing a feature subsampling that ignores position and location of features. For example, a face with the ears in place of the eyes and vice-versa will still be classified as a face by a CNN despite a human clearly recognizing that it is not a face. Finally, CNNs route information in a way that is fundamentally different from how the brain does. While CNNs route all information through all neurons from low to high levels, the brain routes specific information to specialized areas that are better at understanding specific kinds of information.

While computer vision and face recognition have immense potential to revolutionize a number of industries, CNNs are still failing to match the flexibility and accuracy of the human brain. Given a face with the ears and eyes swapped, a CNN will label it as a face despite it clearly appearing to a human as otherwise. This leads us to believe that CNNs only recognize the presence of features and do not do very well recognizing the spatial relationships between features as the brain does so well.

To address the issues with CNNs and in an attempt to build networks that more closely resemble the visual cortex, capsule networks were recently conceived by Hinton et al. In this paper, capsule nets are applied to a face recognition task to explore the workings of this novel network and compare it to the brain.

II. Background

Capsule Networks are a fairly recent invention; there are only three key papers that currently exist on capsule networks authored by Geoffrey Hinton: Transforming Auto-encoders [2], Dynamic Routing Between Capsules [3], and Matrix Capsules with EM Routing [4], which collectively form the definitive pillars outlining the capsule network architecture and the routing algorithm that trains them.

Capsule networks offer a novel approach to image recognition that appears to work in a much more similar manner to the brain by using the spatial relationship of features [5]. Lower layer capsules identify features and include information about the position and location of that feature [1]. After multiplying these lower layer capsules with a weight, the network is able to generate a pose of the face and identify whether it is a face or not. Capsule network architecture attempts to be viewpoint invariant so location and position relationships between features are relevant regardless of the perspective.

The core idea behind capsule networks is the capsule–a group of neurons whose activity vector represents the instantiation parameters of a specific type of entity, such as an object or an object part. In other words, it represents the probability that the entity exists as well as its orientation [3]. The probability of the visual entity being present is locally invariant [2]. Each capsule learns to recognize an implicitly defined visual entity

and then outputs a probability that the entity is present within its limited domain and a set of instantiation parameters such as precise pose, lightning, and deformation. Active capsules at one level make predictions using transformation matrices for the instantiation parameters of higher-level capsules [3]. When multiple predictions agree, a higher level capsule becomes active. This can be further expanded to routing by agreement, where a lower-level capsule prefers to send its output to higher level capsules whose activity vectors have a big scalar product with the prediction coming from the lower-level capsule.

Capsule networks can be seen as an evolution of CNNs in a number of ways. While CNNs use scalar-output feature detectors and max-pooling, capsule networks use vector-output capsules and routing-by-agreement to make predictions [3]. These are backed by biologically plausible models of invariant pattern recognition in the visual cortex. The brain uses dynamic connections and canonical object based frames of reference to generate shape descriptions that can be used for object recognition [6]. In 1993, Olshausen explained how position and scale invariant model of object representation using dynamic routing of information are biologically plausible [7]. While CNNs can have exponential inefficiencies when learning new viewpoints, capsule networks can generalize to novel viewpoints and better understand the intrinsic spatial relationship between features in an image. This is because capsule networks have a more similar structure to the brain. For example, locally invariant probabilities and equivariant instantiation parameters are similar to the output of complex and simple cells respectively [2].

The structure of the brains visual processing network is key to the following results–it uses position and orientation information, it is difficult if not impossible to trick with rearranged features, and it routes information to specialized areas as needed. A capsule network uses similar structures to attempt to achieve the same results. In the brain, there are specific regions or groups of neurons that are used to process different types of visual information and that take into account the configuration of those features. Existing research already shows that inverting a face lowers one's ability to recognize who the face belongs to but does not eliminate a brains ability to recognize that it is a face [8]. Recent research reveals that the brain uses the coordination of a collection of neurons that focus on specific configurations of features to pick out a face [9]. In other words, each face cell, that acts as the highest layer of the network, is primed to a specific ranked combination of facial features to discern what is or is not a face. The brain uses configural processing of faces in three ways–it is sensitive to first-order relations between features to understand that a stimulus is a face, it uses holistic processing to connect features into a gestalt, and it has sensitivity to second-order relations that preserve the distance between features [10]. Brains are capable of recognizing blurred faces with reasonable accuracy since blurring only removes featural information but still allows for holistic processing and sensitivity to second-order relations.

Although face recognition has achieved very high accuracy using convolutional neural networks, such as Googles FaceNet achieving 99.63% accuracy on the Labelled Faces in the Wild dataset [11], these networks are incapable of providing precise spatial relationships between high-level parts due to several stages of subsampling. For instance, if the position of a nose and a mouth are switched on a face, convolutional neural networks will still classify the image as a face. For capsule networks, the two active capsules representing the mouth and the nose have to have the right spatial relationship to activate the next capsule, which in this case will be the face, in order to classify the image as a face. If the predictions for the pose of the face agree for the mouth and the nose, then the mouth and the nose must be in the right spatial relationship to form a face.

Fig. 1. Sample of first augmented dataset

III. Methods

A. Datasets and Sources

Functional implementations of capsule networks exist online that are based on Sabour et als Dynamic Routing Between Capsules paper [3]. For the purposes of this paper, an implementation of a capsule net used to identify traffic signs created by Thibault Neveu was modified for the purposes of facial recognition [12]. This traffic sign implementation is similar to ones created for the MNIST dataset [13]. In the MNIST capsule network, the main structure of the network involves input images being processed by primary capsules that then pass information to secondary digit capsules before classification occurs further down the line [14].

The dataset used was the Labeled Faces in the Wild (LFW) database of face photographs detected by the Viola-Jones face detector [15].

B. Feature Selection & Extraction

Only a portion of the LFW dataset was used, with an initial requirement that each name had a minimum of ten faces. This lead to a total set of 158 unique faces within a dataset of 4324 images. This set was split into two sections: 3459 images for training, and 865 images for testing.

Data augmentation was performed to increase the size of the dataset. Some shearing as well as brightness and contrast shifting was performed. A sample of the augmented dataset can be seen in Figure 1.

A second attempt at training the network used a dataset that required a minimum of 25 faces per person. This lead to a total set of 42 unique faces within a dataset of 2588 images. This set was split into two sections in the same manner as the previous dataset.

Data augmentation was performed to increase the size of the dataset with minor position shifting.

C. Machine Learning Approach

A 3 layer capsule network is used to classify faces for this paper. This is considered a shallow network that uses two convolutional and one fully connected layer. An example of a 3 layer capsule network architecture (used for MNIST classificatin) is shown in Figure 2.

The first convolutional layer converts pixel intensities of the input images to activities of local feature detectors that are used as inputs to the second convolutional layer. The second layer is a convolutional capsule layer, known as the primary capsules. This layer consists of 32 channels of convolutional 8 dimensional capsules. The dimensions can represent features such as hue, position, size, orientation, deformation, texture etc [2]. Each 8 dimensional vector output is sent as input to all the 16 dimensional capsules in the layer above using routing by agreement. The third and final layer has 16 dimensional capsules per class.

Lower level capsules "place-code" information information while high level capsules will "rate-code" positional information in the real-values components of the output vector of a capsule. Place-coding to rate-coding and higher-level capsules represent more complex entities with more degrees of freedom which can be seen in the increasing dimensionality of capsules as one moves up the hierarchy.

D. Training

Dynamic Routing is used between capsules in second and final layer as part of the training process to learn the coupling coefficients, c_{ij}. The routing algorithm is detailed in Algorithm 1.

At the beginning of each routing, all log prior probabilities,b_{ij}, are set to zero. A log prior probability is the probability that capsule i in layer l should be coupled with capsule j in layer $l+1$. The prior probabilities are updated for r iterations to calculate the final output vector of the capsule in the layer $l+1$.

First, the coupling coefficients,c_{ij} between each capsule in layer l and layer $l+1$ are calculated using a routing softmax shown in the Equation 1. This ensures that the coupling coefficients between a capsule in layer l and all the capsules in the layer $l+1$ sum to 1.

$$c_{ij} = \frac{exp(b_{ij})}{\sum_k exp(b_{ij})} \tag{1}$$

Next, the total input,$\boldsymbol{s}_j$ to a capsule j in layer $l+1$ is determined using Equation 2, which is a weighted sum over all prediction vectors, $\hat{\boldsymbol{u}}_{j|i}$. A prediction vector, Equation 3, is the output of a capsule i in layer l multiplied by a weight matrix between capsule i and capsule j.

$$\boldsymbol{s}_j = \sum_i c_{ij} \hat{\boldsymbol{u}}_{j|i} \tag{2}$$

$$\hat{\boldsymbol{u}}_{j|i} = \boldsymbol{W}_{ij} \boldsymbol{u}_i \tag{3}$$

The output vector,$\boldsymbol{v}_{ij}$, of a capsule represents the probability that that entity represented by the capsule is in the current input. This output is calculated using the squash function, Equation 4, which is a non-linearity that ensures short vectors are shrunk to almost zero and long vectors are reduced to slightly below one. Using the agreement between the current output of each capsule j and the prediction vector made by capsule i, the log prior probabilities are updated.

$$\boldsymbol{v}_{ij} = \frac{||\boldsymbol{s}_j||^2}{1+||\boldsymbol{s}_j||^2} \frac{\boldsymbol{s}_j}{||\boldsymbol{s}_j||} \tag{4}$$

For the purposes of the report, an existing architecture developed by Thibault Neveu was primarily leveraged, which was implemented in Tensorflow[12]. The final model was trained with GPUs on Floydhub for two hours, as that was the maximum time allocated to allow the model to run. The final model was trained for 2000 steps and was terminated after that since the training error had stagnated at 100%, and there were clear signs of overfitting as discussed in the following section.

The hyperparameters of batch size and learning rate were chosen as 128 and 0.001 based on the parameters used in the capsule network described in *Dynamic Routing Between Capsules* [3]. The hyperparameter of min_faces_per_sample was set to 25 in order to balance the size of the data set with the number of faces per label.

E. Error Metrics

The goal is to have top-level capsules for each unique face class k with a long instantiation vector if and only if the predicted face is in the image. This results in the following equation:

$$L_k = T_k max(0, m^+ - ||\boldsymbol{v}_k||)^2 + \lambda(1-T_k) max(0, ||\boldsymbol{v}_k|| - m^-)^2 \tag{5}$$

$T_k = 1$, $m^+ = 0.9$ and $m^- = 0.1$ if and only if face of class k is present. To stop the initial learning from shrinking the lengths of the activity vectors of all the face capsules, λ, the down-weighting of the loss for absent face classes, is used. The total loss is the sum of the losses for all the face name capsules. The sum of margin losses is minimized using the Adam optimizer.

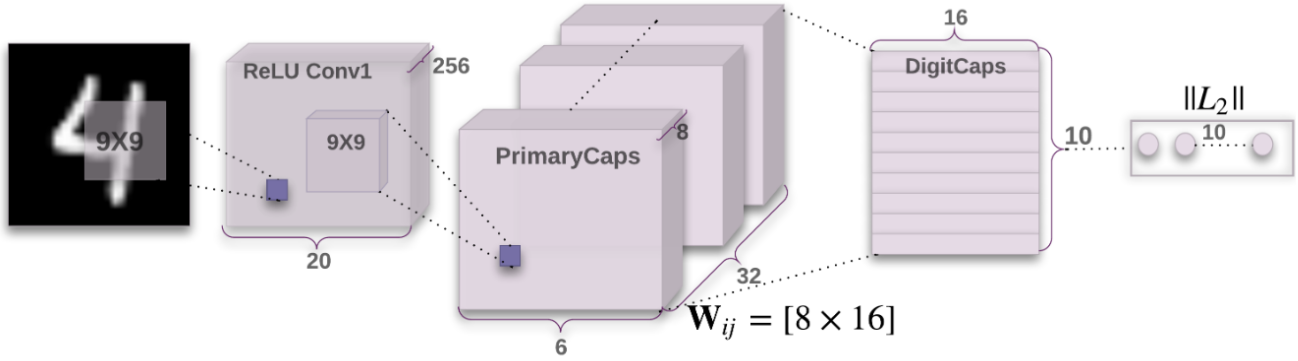

Fig. 2. Capsule Network Architecture [3]

Algorithm 1 Routing algorithm [3]

procedure ROUTING($\hat{\boldsymbol{u}}_{j|i}, r, l$)
  for all capsule i in layer l and capsule j in layer $l+1$:
  $b_{ij} \leftarrow 0$
  for r iterations **do**
    for all capsule i in layer l: $\boldsymbol{c_i} \leftarrow \texttt{softmax}(\boldsymbol{b_i})$ ▷ `softmax` computes Eq. 3
    for all capsule j in layer $(l+1)$: $\boldsymbol{s}_j \leftarrow \sum_i c_{ij} \hat{\boldsymbol{u}}_{j|i}$
    for all capsule j in layer $(l+1)$: $\boldsymbol{v}_j \leftarrow \texttt{squash}(\boldsymbol{s_j})$ ▷ `squash` computes Eq. 1
    for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow b_{ij} + \hat{\boldsymbol{u}}_{j|i} \cdot \boldsymbol{v}_j$
  return $\boldsymbol{v}_j$

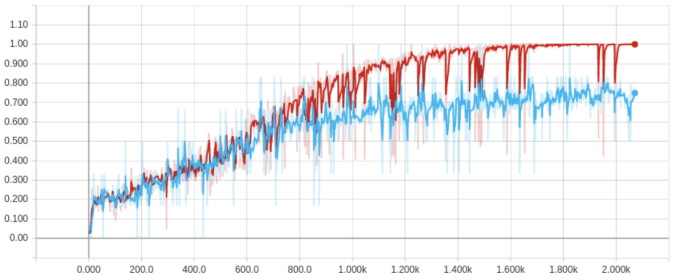

Fig. 3. Graph comparing training accuracy (red) versus validation accuracy (blue)

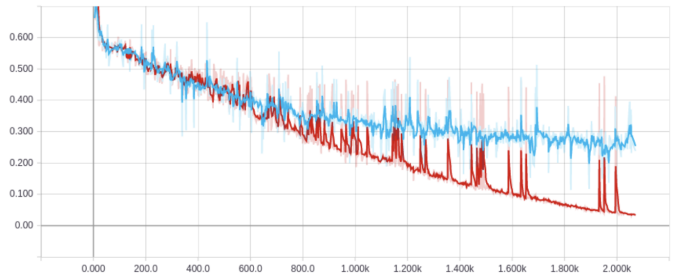

Fig. 4. Graph comparing training loss (red) versus validation loss (blue)

IV. RESULTS

After training the capsule network, both accuracy and loss were recorded for training and validation datasets.

Some things of note–the final loss on the entire test dataset for the final model was 0.088 while the final accuracy on entire test dataset is 93.7%. This is significant as capsule networks trained and validated on the various MNIST datasets had accuracies around 99.2% depending on the architecture and dataset used [3].

The brain is capable of identifying inverted faces without being trained on inverted data as a result of the way visual processing is structured. In the same way, testing the capsule network with an inverted face revealed that the network was able to identify the inverted face without additional training.

After the model was trained, an inverted face was passed in alongside a regular face to compare the outputs, seen in Figure 5. The discrepancy in the highest and second highest softmax layer being lower for the inverted faces corresponds to biological data, where human beings in general have a tougher time recognizing inverted faces than non-inverted ones. An important factor to note in the above inversion results is that while these are some examples that show the networks are able to recognize inverted results, as seen in the Figure 5, the models accuracy is significantly lower as a whole on inverted faces.

Current literature on capsule networks discusses how "CapsNets [are] moderately robust to small affine transformations [3]. This claim was tested with this facial recognition capsule network. The results are shows in Figure 6.

From Figure 6, there is a sharp decrease in accuracy in the network after a 30 degree rotation. At a rotation of 5 degrees, the accuracy of the network is 93.14%. This is comparable and even better than some networks described in existing literature. CapsNet was able to achieve an accuracy of 79% on the affnist dataset, which is an augmented dataset of the MNIST dataset where the original dataset has been transformed [3]. A

Fig. 5. Comparing the results of a regular and inverted face through the trained model.

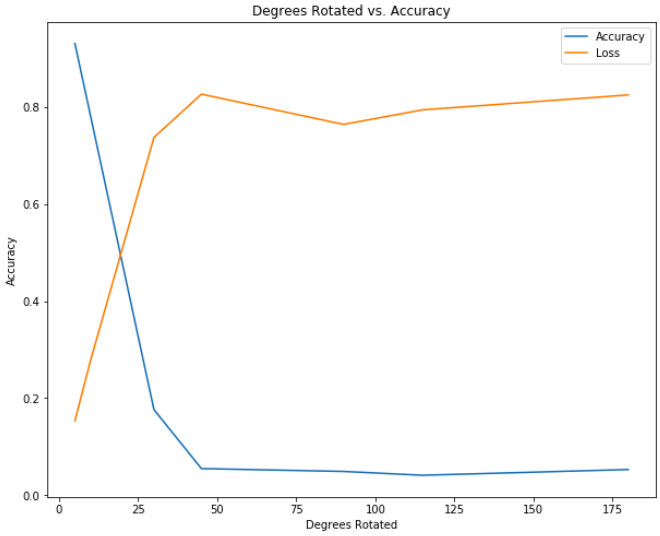

Fig. 6. Graph of test accuracy as images are rotated

CNN was only able to achieve 66% accuracy on the affnist dataset [3]. Similar results are demonstrated on other datasets outlining the superior performance of CapsNets over CNN in recognizing unseen viewpoints [3]. This is an important fact, as it illustrates how capsule networks handle rotations better than CNNs that likely results from the capsule networks understanding of the spatial relationships between features that is used for prediction.

To further test the invariance of CapsNets to unseen data, the test dataset was transformed with a Gaussian blur and run through the network to produce the accuracies seen in Figure 7. Even with a high amount of blur, the CapsNet perform quite well and is comparable to the state of the art CNNs which are much deeper and trained with much more data.

Together, these results indicate that capsule networks perform better at facial recognition than CNNs when faced with transformed face images. High levels of noise, blur, missing pixels, and brightness have a detrimental effect on the verification performance of all [CNN] models [16]. The current wisdom for CNNs to generalize to different viewpoints requires lots of data and leverages maxpooling, which only provides small translational invariance. 2D Convolution is equivariant under translation but not rotation, and this allows capsule networks to outperform CNNs when dealing with

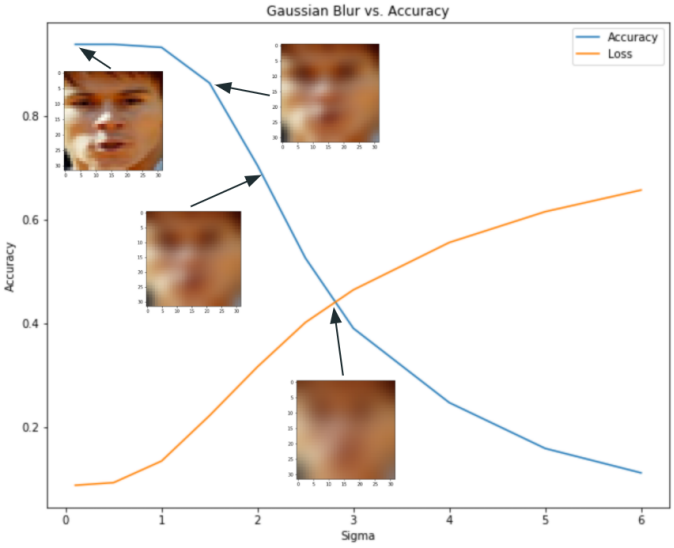

Fig. 7. Graph of test accuracy as images are blurred

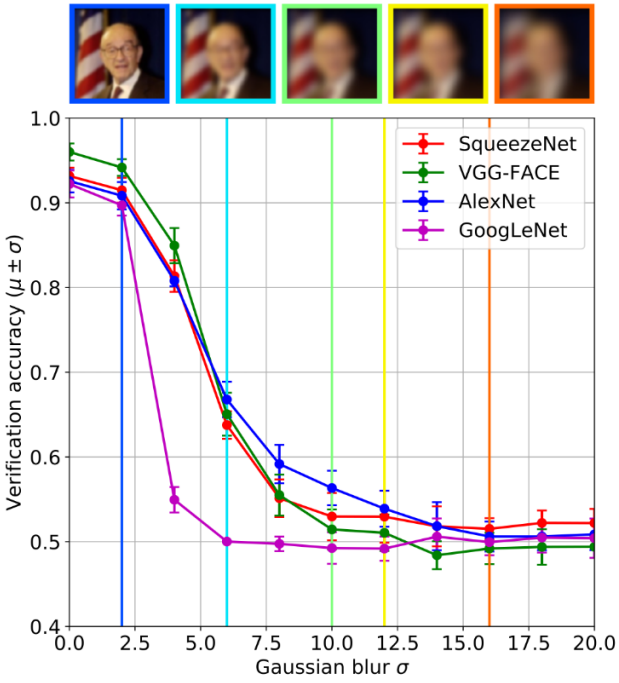

Fig. 8. Performance of State-Of-The-Art CNNs on blurred faces [16]

rotation and proportion change. Capsule networks have more equivariance properties and so require less data and fewer layers–which is good news as both data and computational power can be prohibitively expensive to most. Lastly, capsule networks are also thought to be more robust to adversarial attacks [4].

V. Conclusions & Recommendations

"An under-trained CapsNet with early stopping which achieved 99.23% accuracy on the expanded MNIST test set achieved 79% accuracy on the affnist test set. A traditional convolutional model with a similar number of parameters

which achieved similar accuracy (99.22%) on the expanded MNIST test set only achieved 66% on the affnist test set." [3]. It is clear that the performance of the capsule network trained on the small segment of the LFW dataset is comparable. From the accuracy and loss graphs, the facial recognition model seems to be overfitting. This is likely due to a lack of data. Additionally, the number of routing iterations is an important factor in the performance of the capsule network. Increasing routing iterations tend to increase network capacity and overfit the training dataset [3].

For future experiments on face recognition using capsule networks, experimenting with fewer iterations of routing, larger training set size and larger image size could improve the overall accuracy and avoid overfitting. The number of iterations of routing should be experimented with to verify the convergence of b_{ij}, the log prior probability. As the number of iterations increase, b_{ij} improves, however, increasing iterations also increases the network capacity, which is undesirable. Therefore, the optimal tradeoff between an additional iteration and change in b_{ij} should be found. Furthermore, a larger training set with more samples for each face can also avoid overfitting. The image size was reduced to 32 by 32, however, the downsampling resulted in a loss of resolution. Training the model on larger images and better resolution will definitely increase accuracy, as there would be more information for the network to better discriminate and learn different faces appropriately. Overall this approach suffers from the same problems as most other machine learning tasks: a larger and more robust dataset to better suit the network architecture is required.

Finally an important step to consider for more accurate comparisons would be to build an affine transform dataset for faces to provide a direct comparison with established results. In addition, when comparing results between established literature and face recognition results, further analysis has to be done comparing the response activities of the different capsule neurons between the MNIST dataset and the face dataset, since recognizing MNIST characters is considered to be an easier task compared to face recognition.

Despite the overfitting, the results of the capsule network on blurred and inverted faces are very promising. Clearly, capsule networks have the potential to recognize inverted faces without requiring any training on inverted faces beforehand, just like the brain. Furthermore, the capsule networks performance in recognizing blurred images provides some confirmation that the use of spatial relationships creates accuracy comparable to state-of-the-art networks even with a significantly more limited training dataset.

From early literature on capsule networks and the results of this paper, it can be concluded that capsule networks outperform CNNs on unseen transformed datasets.

REFERENCES

[1] G. Hinton, "What is wrong with convolutional neural nets? - Brain & Cognitive Sciences - Fall Colloquium Series", MIT, 2014.

[2] G. Hinton, A. Krizhevsky and S. Wang, "Transforming Auto-Encoders", Lecture Notes in Computer Science, pp. 44-51, 2011.

[3] S. Sabour, N. Frosst and G. Hinton, "Dynamic Routing Between Capsules", Advances in Neural Information Processing Systems 30 (NIPS 2017), 2017.

[4] G. Hinton, S. Sabour and N. Frosst, "Matrix capsules with EM routing", ICLR 2018 Conference, 2018.

[5] N. Bourdakos, "Capsule Networks Are Shaking up AI–Heres How to Use Them", Hacker Noon, 2017.

[6] G. Hinton, "A parallel computation that assigns canonical object-based frames of reference", Proceedings of the 7th international joint conference on Artificial intelligence, vol. 2, no. 1981, pp. 683-685, 1981.

[7] B. Olshausen, C. Anderson and D. Van Essen, "A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information", Journal of Neuroscience, vol. 13, no. 11, pp. 47004719, 1993.

[8] B. Rossion and I. Gauthier, "How Does the Brain Process Upright and Inverted Faces?", Behavioral and Cognitive Neuroscience Reviews, vol. 1, no. 1, pp. 63-75, 2002.

[9] L. Chang and D. Tsao, "The Code for Facial Identity in the Primate Brain", Cell, vol. 169, no. 6, pp. 1013-1028.e14, 2017.

[10] D. Maurer, R. Le Grand and C. Mondloch, "The many faces of configural processing", TRENDS in Cognitive Science, vol. 6, no. 6, pp. 255-260, 2002.

[11] F. Schroff, D. Kalenichenko and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering", 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[12] T. Neveu, "Capsnet - Traffic sign classifier - Tensorflow", GitHub, 2018. [Online]. Available: https://github.com/thibo73800/capsnet-traffic-sign-classifier.

[13] A. Gron, How to implement CapsNets using TensorFlow. YouTube, 2017.

[14] A. Geron, "Capsule Networks (CapsNets)", GitHub, 2018. [Online]. Available: https://github.com/ageron/handson-ml/blob/master/extra_capsnets.ipynb.

[15] "LFW Face Database", University of Massachusetts, 2018. [Online]. Available: http://vis-www.cs.umass.edu/lfw/.

[16] K. Grm, V. truc, A. Artiges, M. Caron and H. Ekenel, "Strengths and weaknesses of deep learning models for face recognition against image degradations", IET Biometrics, vol. 7, no. 1, pp. 81-89, 2018.