NORTHWESTERN UNIVERSITY

Multimodal Data Fusion and Feature Visualization in Convolutional Neural Networks

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Electrical and Computer Engineering

By

Arjun Naresh Punjabi

EVANSTON, ILLINOIS

June 2020

ABSTRACT

Multimodal Data Fusion and Feature Visualization in Convolutional Neural Networks

Arjun Naresh Punjabi

Convolutional neural networks have become a staple in computer vision and image processing tasks. The capacity for these networks to perform visual pattern recognition in a data-driven fashion has prompted explosive growth in a myriad of applications. That said, despite their popularity, there are still facets of these networks that merit further investigation. This dissertation will describe two such directions. The first is related to multimodal data fusion, in which a network takes in multiple sources of data. The specific application in this instance is medical imaging, and the investigation gives insight into the relative efficacies of each data type in a traditional classification setting as well as a regression and longitudinal prediction scenario. The second part of the dissertation concerns deep visualization. This concept attempts to develop understanding of neural networks through the generation of descriptive images. Here, these techniques are applied to a variety of networks using standard computer vision datasets. Fundamentally, the multimodal fusion study highlights the potential power of the convolutional neural network, while the deep visualization study develops intuition and interpretability of these often obfuscated algorithms.

Acknowledgments

I would like to dedicate this dissertation to my parents and grandfather, whose support was vital to my success, even when I didn't think I needed it. I would like to thank my advisor, Professor Aggelos Katsaggelos, for being a friend and mentor during my graduate school experience. I would also like to thank my other committee members, Todd Parrish and Oliver Cossairt, for their contributions. Finally, I would like to thank some of my colleagues who have been a part of my work in a variety of ways. This includes, but is not limited to: Alice Lucas, Emanuel Azcona, Semih Barutcu, Arun Sankisa, Qiqin "Tim" Dai, Pablo Ruiz Mataran, Juan Gabriel Serra Perez, Henry Chopp, Amit Adate, and Jonas Schmid.

Table of Contents

CHAPTER 1

Background on Multimodal Data Fusion

Multimodal data fusion is the the process by which several data acquisition streams, or modalities, are aggregated in a single algorithm. The motivation for this approach arises from the concept that different sources of data may contain complementary pieces of information. As a result, an algorithm can leverage the specific strengths of each modality to improve performance [**48**]. In other words, the fusion of multiple modalities may be more than the sum of their parts.

That said, fusion is not a simplistic exercise. One must take care when fusing multiple data modalities, as incongruous data types can negatively impact performance. Typically, one should fuse data that have complementary, but not redundant, information. One of the most common examples is fusing audio and video data. One could, for instance, create a speech recognition algorithm that takes as input audio recordings and corresponding video of the speaker. The video data supplements the audio recording and should improve the ability for the classifier to correctly identify the word or sound. That said, if the disparity between the two data types is high (i.e. the audio and video data support different classifications), the fusion algorithm may actually perform worse than an algorithm that takes only audio or only visual data. While the phenomenon has been experimentally demonstrated with several types of algorithms [**43**], there is also justification in human cognition experiments. The "McGurk effect", demonstrated in [**61**], showed that humans will perceive entirely different syllables when presented with two contradictory syllables from the audio and visual data.

Fundamentally, one must critically examine all available data modalities before fusion in order to ascertain the possible gain in efficacy from their combination.

Additionally, the methodology for fusing two data modalities can impact performance. Beyond the technicalities that vary between algorithms, there is a general principle of when to fuse the data within the pipeline. One can typically make a distinction between early fusion or late fusion; that is, whether the modalities are combined before the majority of processing or after. This distinction can greatly influence performance, as in [**92**]. Moreover, the results in [**92**] do not conclusively determine whether one scheme is superior. Rather, this decision may need to be made on a case by case basis.

All of the aforementioned concepts are noticeably apparent in the medical domain, where various forms of patient data are collected in a battery of scans and examinations. In some sense, doctors and medical professionals are constantly performing their own data fusion as they absorb several patient data modalities in order to make clinically relevant decisions. A challenging and impactful problem arises when a convolutional neural network is used in this manner. The following chapters will examine the specific fusion of MRI and PET data with the goal of diagnosing Alzheimer's disease. While some of the results of the investigation are specific to this application, much of the intuition is also informative in the study of data fusion as a whole.

CHAPTER 2

Neuroimaging Modality Fusion in Alzheimer's Classification Using Convolutional Neural Networks

2.1. Chapter Abstract

Automated methods for Alzheimer's disease (AD) classification have the potential for great clinical benefits and may provide insight for combating the disease. Machine learning, and more specifically deep neural networks, have been shown to have great efficacy in this domain. These algorithms often use neurological imaging data such as MRI and FDG PET, but a comprehensive and balanced comparison of the MRI and amyloid PET modalities has not been performed. In order to accurately determine the relative strength of each imaging variant, this work performs a comparison study in the context of Alzheimer's dementia classification using the Alzheimer's Disease Neuroimaging Initiative (ADNI) dataset with identical neural network architectures. Furthermore, this work analyzes the benefits of using both modalities in a fusion setting and discusses how these data types may be leveraged in future AD studies using deep learning.

2.2. Introduction

Alzheimer's disease (AD) is a neurodegenerative disorder characterized by cognitive decline and dementia. The number of individuals living with AD in the United States is expected to reach 10 million by the year 2025 [**31**]. As a result, automated methods for computer aided diagnosis could greatly improve the ability to screen at-risk individuals.

Such methods typically take as input patient data including demographics, medical history, genetic sequencing, and neurological images among others. The resulting output is health status indicated by a diagnosis label, which may also include a probabilistic uncertainty on the prediction. This particular investigation will focus on two different neuroimaging modalities: structural T1-weighted MRI and AV-45 amyloid PET. The primary goal of the investigation is to compare the efficacy of each of these modalities in isolation as well as when both are used as simultaneous input to a fusion system. While other studies make use of T1-weighted MRI and FDG PET, we believe that, to the best of our knowledge, this is the first comparison and fusion deep learning study using AV-45 amyloid PET. Because FDG and amyloid PET have different biological sources, their ability to aid in Alzheimer's diagnosis may greatly differ.

The algorithmic design of these methods can vary, but recent successes in machine learning have opened the floodgates for a plethora of deep neural networks trained for computer aided diagnosis. Given the visual nature of the input data, this work opted to apply a model well suited for computer vision tasks: the convolutional neural network (CNN). The following sections will focus on related approaches to the AD classification problem, the methodology of both the network and data pre-processing pipeline, and a discussion of the classification results.

2.3. Related Work

Computer aided diagnosis methods in this domain have spanned the gamut of algorithmic design. Earlier methods often applied linear classifiers like support vector machines (SVM) to hand-crafted biological features [**14**]. These features can be defined at the individual voxel level, as in the case for tissue probability maps, or at the regional level, including

cortical thickness and hippocampal shape or volume. The 2011 comparison performed in [**14**] found that whole brain methods generally achieved higher classification accuracy than their region-based counterparts. Additionally, there was evidence to suggest that certain data pre-processing methods, namely the DARTEL registration package [**5**], can substantially impact classification results. These two findings informed the decision to use whole brain volumes in this work and design a robust registration pipeline before the classification algorithm.

Similar linear classifier or SVM-based methods exist that align with these ideas. In [**46**], gray matter tissue maps were classified with an SVM. A more complex scheme exists in [**57**], where template selection was performed on gray matter density maps and these features were clustered in preparation for SVM classification. As previously discussed, regional features can also be used as input to an SVM, such as spherical harmonic coefficients calculated from the hippocampus [**23**]. In [**78**], the analysis is extended to other linear classifiers, primarily comparing the performance between SVMs and variations of random forest classifiers on a large conglomerate of Alzheimer's datasets. These models can also extend to multiple data modalities as in [**118**], where features from MRI and FDG PET data were extracted and combined with a kernel-based approach. In [**116**], the procedure was modified with a custom loss function in order to perform both diagnosis classification and cognitive score regression simultaneously using a modified support vector-based model trained with MRI, PET, and cerebrospinal fluid (CSF) images.

Despite the initial popularity of SVMs and linear classifiers, there has been a transition in the last several years toward more non-linear approaches. Namely, the introduction of artificial neural networks has transformed the landscape of automated Alzheimer's dementia diagnosis. However, even these methods have varied in construction. The works in [**94, 93**] used a deep Boltzmann machine (DBM) to extract features from MRI and FDG PET data

which are then classified using an SVM. Similarly, a DBM was also used in [**53**] to extract features from MRI and FDG PET, but additionally included CSF and cognitive test scores. The features are still classified with an SVM. A more standard fully-connected neural network was trained on MRI images in [**106**], but performance was improved by adding spatial neighborhood regularization similar to the receptive field of convolutional kernels.

This leads to the current preferred machine learning model, the CNN. These models are well suited to tasks with 2D or 3D data due to the shared filter weights within each convolutional layer. A CNN was proposed in [**81**] that takes fMRI slices as input to a modified LeNet-5 CNN architecture [**52**]. The DeepAD paper [**82**] further developed this notion by utilizing the more complex GoogleNet CNN [**95**]. In [**54**], MRI and FDG PET data were used to train a multimodal CNN for classification, but it also allowed for missing modalities and modality completion. Some methods opted to use autoencoders [**33**] which can employ convolutional filters, but structurally differ from CNNs. While CNNs are trained to map input images to some given representation, autoencoders are trained to perform dimensionality reduction and reconstruct the input image. In this manner, the features learned in the middle layer of an autoencoder can be extracted and classified with either linear or non-linear methods. In [**58**], features from MRI and FDG PET images were extracted using a stacked autoencoder which were then classified with softmax regression. On the other hand, the work in [**29**] used an autoencoder on 2D MRI slices to learn basis features that are then used as CNN filters. A similar procedure was performed in [**71**] that compared the performance between both 2D and 3D systems. An autoencoder was used in [**34**] on full 3D MRI images to pre-train the layers of a CNN model, and this was expanded in [**101**] to include the FDG PET modality. The authors in [**87**] use a scheme of stacked polynomial

networks on MRI and FDG PET data, and use similar cascaded network approaches in [**86**] and [**25**] when tackling Parkinson's diagnosis. Some of these results are shown in Table 2.1.

	MRI	FDG PET	Fusion
[**93**] DBM	92.38	92.20	95.35
[**101**] CNN	80.62	81.93	84.72
[**101**] SAE	85.24	85.53	91.14
[**87**] SDPN	95.44	95.11	97.13

Table 2.1. MRI and FDG PET Fusion Classification Accuracies (%)

Fundamentally, while methods exist that take advantage of multiple data types and apply state-of-the-art neural network architectures, comparison studies between modalities have been haphazard in their use of datasets and lacking in explanations of model efficacy. In some instances, subsets of larger databases were used without explanations of why certain images were included or excluded. The deep learning comparisons that have been performed examine MRI and FDG PET scans, whereas none have addressed fusion of MRI with AV-45 PET scans. Because FDG PET measures metabolism whereas AV-45 PET measures beta amyloid (the buildup of which is a precursor to Alzheimer's disease), the modalities are drastically different in their information content [**39**]. Consequently, the added benefit to classification performance when combined with MRI data may differ as well. Additionally, pre-processing pipelines differ between these various studies. These factors contribute to incongruous modality comparison results between papers. Furthermore, the biological explanations for such discrepancies are often lacking or non-existent. This work is novel in these respects. First, the pre-processing used in this work is clearly explained and the rationale for each step is provided. Also, the modality comparison results are discussed within a biological context that more effectively describes the relative performance of each data type.

2.4. Methodology

As previously alluded to in the discussion of related work, pre-processing operations can have a major impact on final classification performance. As a result, a pipeline was developed to correct several of the biases inherent in the imaging data. While the components of the pipeline employ existing algorithms, the overall structure differs from previous work and allows for a more fair comparison between the T1-weighted MRI and AV-45 PET modalities.

This section also discusses the neural network architecture. The design of the network is similar to the CNN-based approaches discussed previously. Again, because the primary goal of the investigation is a comparison of data modalities rather than network styles, the CNN was designed to be representative of comparable methods comprised of standard network layers.

2.4.1. Pre-processing

The pre-processing pipeline aimed to correct several biases that can exist in raw MRI and PET data. This also removes the additional burden of the network learning methods to correct or overlook these biases. Instead, the network has the isolated task of finding patterns between healthy and Alzheimer's patients. The vast majority of related work also employs similar pre-processing techniques in order to combat standard problems; namely, most methods perform some kind of MRI bias field correction, volumetric skull stripping, and affine registration. This approach is nascent in its registration scheme in order to prepare data for longitudinal studies in addition to traditional single time instance analyses. This manifests itself in two ways. First, our current investigation that treats each of these scanning instances as distinct samples in the dataset is less biased by differences in pre-processing for

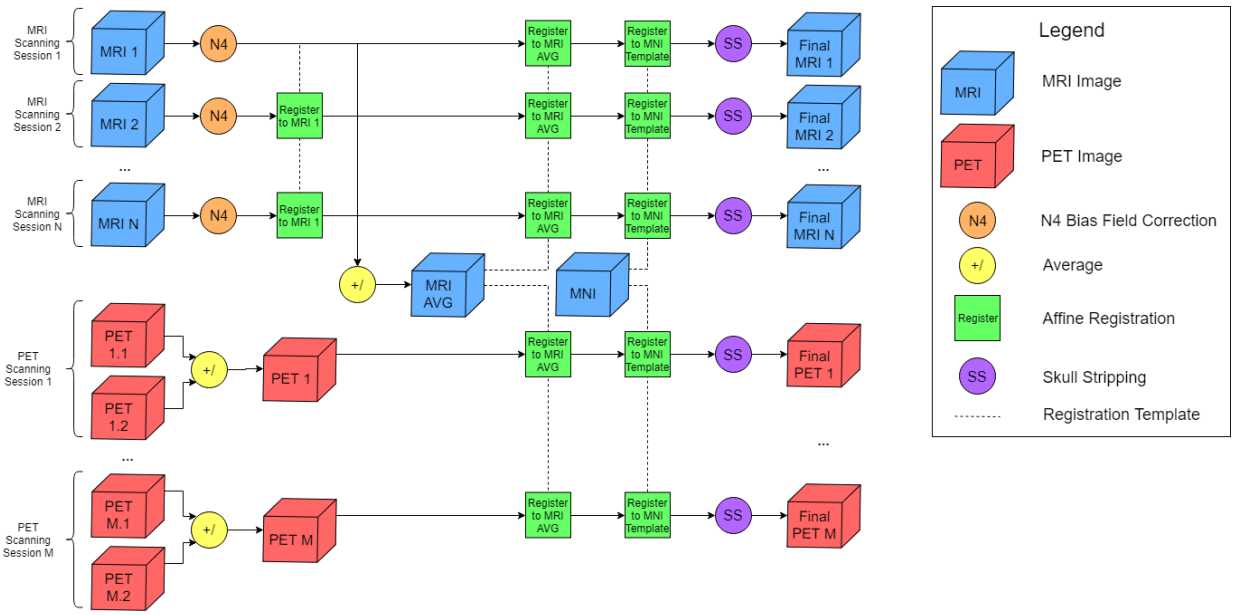


Figure 2.1. Pre-processing pipeline for a single subject. A subject has N MRI scanning sessions and M PET scanning sessions; therefore, the pipeline yields N MRI images and M PET images. The pipeline is repeated for each subject in the dataset.

each modality. Second, when the scanning instances are viewed jointly as a single sample in the dataset for a longitudinal study, the images are normalized both within the subject and among all subjects in the set. Future longitudinal studies that take advantage of this processed data will be discussed at the end. The building blocks of the pipeline are as follows:

2.4.1.1. MRI Bias Field Correction. MRI images can have a low frequency bias component as a result of transmit/receive inhomogeneities of the scanner [**62**]. This spatial non-uniformity, while not always visually apparent, can cause problems for image processing pipelines. As a result, many MRI processing schemes begin by applying a bias field correction algorithm. Non-parametric non-uniform intensity normalization (N3) [**90**] is a robust and well-established approach for removing this bias field. It optimizes for the slowly varying multiplicative field that, when removed, restores the high frequency components of the true

signal. This work opted to employ a more recent update to this method known as N4 [**99**], which makes use of B-spline fitting for improved corrections. This step is performed on the raw MRI images and is unnecessary for the PET images.

2.4.1.2. Affine Registration. Both image modalities are registered using a linear affine transformation. Registration aims to remove any spatial discrepancies between individuals in the scanner, namely minor translations and rotations from a standard orientation. Typically, scans are registered to a brain atlas template, such as MNI152 [**21**]. While this procedure is perfectly acceptable for traditional single time point analyses, this pipeline was designed to accommodate longitudinal studies as well. In such a setting, a patient in the dataset will have multiple scanning sessions at different times, but these images are aggregated and treated as a more complex representation of a single data point. As a result, it is beneficial to have congruence between the temporal scans in addition to registration to the standard template. Consequently, MRI and PET scans in the pipeline are registered first to an average template created from all MRI scans from a single patient, and then once more to the standard MNI152 space. The average template is created by registering all scans from one patient to a single scanning instance and then taking the mean of these images. Therefore, each subject will have unique average templates. Every MRI and PET scan is registered to the respective average template before the traditional registration with the MNI152 template. This ensures that all of the scans are registered both temporally within each patient's history and generally across the entire dataset. FSL FLIRT was used to perform the registrations [**41**].

2.4.1.3. Skull Stripping. Skull stripping is used to remove non-brain tissue voxels from the images. This is generally framed as a segmentation problem wherein clustering can be used to separate the voxels accordingly, as in FSL's brain extraction tool (BET) [**91**].

However, given that the scans were already registered to a standard space, skull stripping was a straightforward task. A brain mask in MNI152 space was used to zero out any non-brain voxels in both the MRI and PET images.

Fig 2.1 shows the pipeline in its entirety. The process is performed for all MRI and PET images for a single patient in the dataset before proceeding to the next. N4 correction is applied to all of the MRI scans before any registration steps. All MRI scans are registered to the first scanning time point, and the resulting images are averaged to create the average template. The N4 corrected scans are registered to this space before being registered with the MNI152 template. The resulting images are then skull stripped using a binary mask.

Amyloid AV-45 PET scans were collected over 20 minutes in dynamic list-mode 50 minutes post-injection of 370 MBq 18F-florbetapir. PET scans were attenuation corrected using a computed tomography scan. The first 10 minutes of PET acquisition was reconstruction into two 5 minute frames. Frames were motion corrected together and referenced (normalized) by the whole cerebellum. Each PET scan was registered to the individual's average T1 template with a 6 DOF registration and then the pre-computed 12 DOF registration from average T1 to MNI152 was concatenated and applied to the PET images to move them from native PET to MNI152 space. Finally, the PET images were skull stripped as above.

2.4.2. Network

The CNN architecture is fairly traditional in its construction and is most similar to that in [**34**]. Because the goal of this investigation is modality comparison, a representative CNN architecture was used rather than one with very specific modifications aimed at maximizing classification scores. In this manner, the modality comparison would not be obfuscated by the nuances of the network. The network takes as input a full 3D MRI or PET image and

outputs a diagnosis label. While several processing layers exist in the network, there are only three different varieties: convolutional layers, max pooling layers, and fully connected layers. Convolutional layers constitute the backbone of the CNN. As the name suggests, 3D filters are convolved with the input to the layer. Each kernel is made of learned weights that are shared across the whole input image; yet, each processing layer can have multiple trainable kernels. This allows kernel specialization while still affording the ability to capture variations at each layer. Following convolutional layers, it is common to have max pooling layers. These layers downsample an input image by outputting the maximum response in a given region. For example, a max pooling layer with a kernel size of 2x2x2 will result in a output image that is half the input size in each dimension. Each voxel in the output will correspond to the maximum value of the input image in the associated 2x2x2 window. Fully connected layers are often placed at the end of a CNN. These layers take the region specific convolutional features learned earlier in the network and allow connections between every feature. The weights in these layers are also trainable; therefore, these layers aggregate the region features and learn global connections between them. As a result, the output of the final fully connected layer in the CNN is the final diagnosis label.

Figure 2.2 is a diagram of the final CNN architecture for a single modality. In this instance, the network accepts MRI or PET images of size 182x218x182 (due to the MNI template size), but in principle a CNN can accept an image of any size. The image is then processed by three pairs of alternating convolutional (20 kernels of size 5x5x5) and max pooling layers (kernel size 2x2x2). The convolutional layers use the ReLU [**30**] activation function. Following these layers, the feature vector is flattened before being passed as input to a fully connected layer with 1024 nodes, a second fully connected layer of 128 nodes, and finally a fully connected layer with the number of diagnosis categories. In this case, there

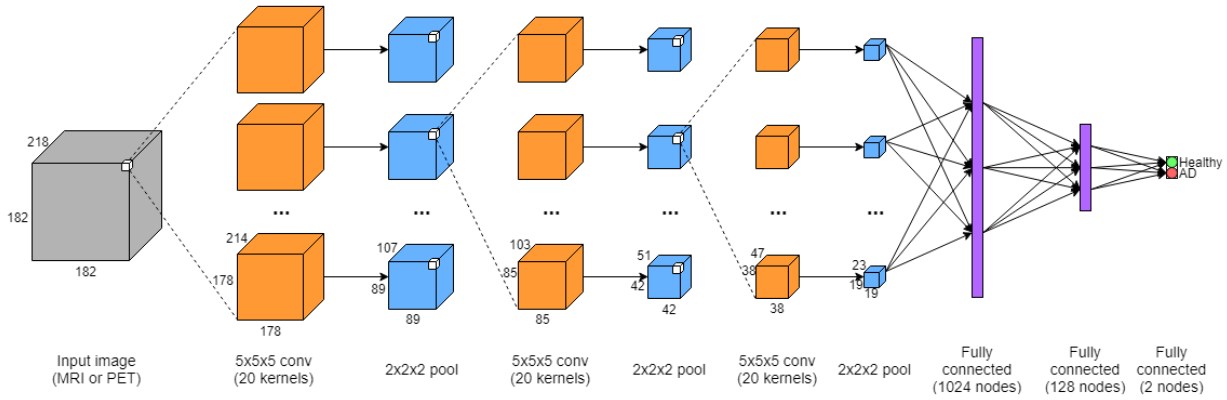


Figure 2.2. Convolutional neural network for one modality. A single MRI or PET volume is taken as input, and the output is a binary diagnosis label of either "Healthy" or "AD".

are 2 diagnosis categories corresponding to individuals with AD and healthy controls. The two fully connected layers also use the ReLU activation function, but the final classification is done with the softmax function.

Figure 2.3 shows the extension of the network for the fusion case. In this setting, the network takes both an MRI and PET image of size 182x218x182 as input into parallel branches. These branches are structured in the same manner as in the former case, but an additional fully connected layer of 128 nodes is added at the end in order to fuse the information from both modalities before the final classification is made. Additionally, the number of kernels in each convolutional layer was changed from 20 to 10 in order to keep the number of weights in the fusion network approximately the same as in the single modality network.

2.5. Experimental Design

Classification experiments were performed on the Alzheimer's Disease Neuroimaging Initiative (ADNI)[**38**] database. The primary goal of ADNI has been to test whether serial MRI,

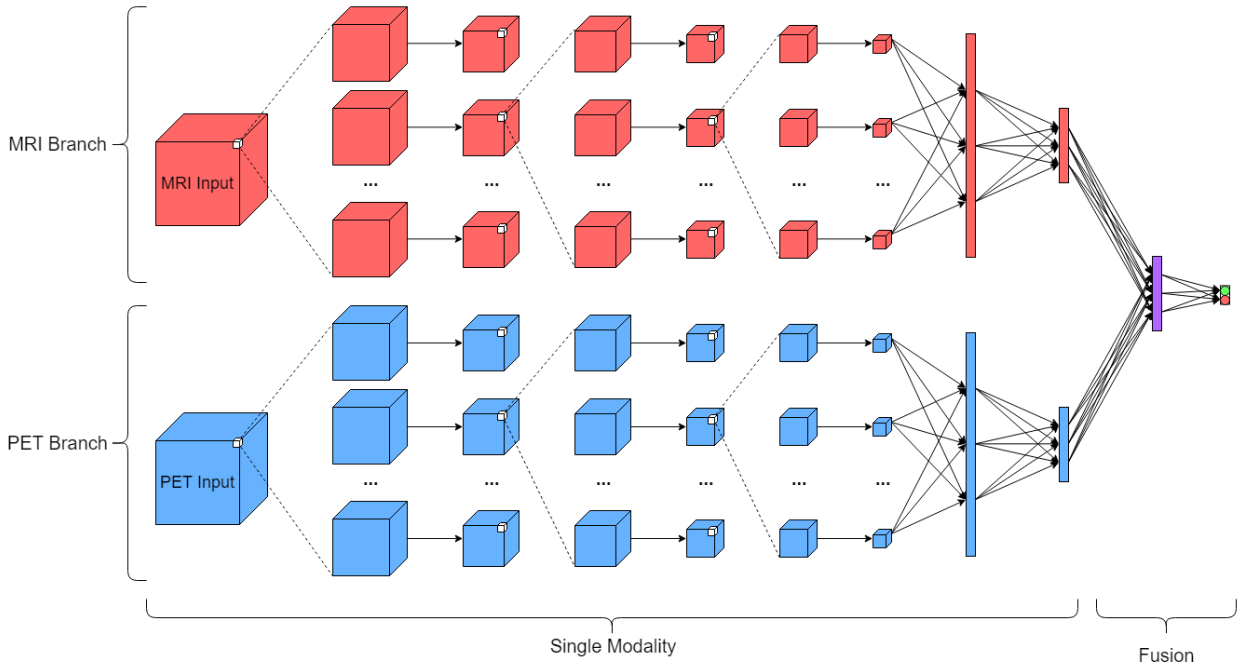


Figure 2.3. Convolutional neural network for fusing MRI and PET modalities. An MRI and PET scan from a single patient is taken as input, and the output is again a binary diagnosis label.

PET, other biological markers, and clinical and neuropsychological assessment can be combined to measure the progression of mild cognitive impairment (MCI) and early Alzheimer's disease. The set has clinical data from hundreds of study participants including neuroimaging modalities, demographics, medical history, and genetic sequencing. This work analyzed T1-weighted MRI and amyloid PET images in addition to the diagnosis labels given to patients at each study visit. Neurological test scores were examined in order to validate these labels, but were not used during network training. Data was only used from participants who had at least one scanning session for both MRI and PET. Additionally, scanning sessions were not considered if neurological testing was not performed within 2 months of the scanning session. This was to ensure that the diagnosis label provided during the scanning sessions had clinical justification. As a result, a subset of 723 ADNI patients were used.

As in [**82**], individual scanning sessions from the same patient were considered separately in this work. This resulted in 1299 MRI scans, each falling into either the healthy or AD category. Patients underwent less PET scanning, with a total of 585 scans. Classification experiments were initially performed using only one modality, either MRI or PET, and using the appropriate data subset. Due to the fact that more MRI data exists than PET data, two different MRI classification experiments were performed. In one case, all of the available MRI data was used. In the other, the MRI data was limited to only use the same number of scans as the PET dataset.

These sets were further split into training and testing components in order to ascertain the generalizability of the algorithm. When splitting the data into training and testing subsets, scanning sessions from a single patient were not used in both the testing and training subsets. In other words, all of a single patient's scans were used in one of the two subsets. This was done to ensure that the algorithm would not overfit to the patient's identity rather than learning the disease pattern. In some previous works, it is unclear whether this procedure was done. As a result, classification results in some previous work may have been inflated by models that overfit on certain individuals in the dataset.

Following this, fusion experiments were performed, where an MRI and PET scan from the same individual at a given time were used. Each scan was sent through parallel CNN branches. At the final fully connected layer of each branch, the features were merged into another fully connected layer that was used to produce the classification result. These experiments used the same number of data points as the PET experiments, albeit with each data point having an associated MRI and PET scan. Again, the testing and training subsets were made such that no patient's data was used in both subsets.

The neural network was constructed in Python using Keras [**12**] as a front-end and Tensorflow [**1**] as the back-end deep learning framework. The optimization procedure used stochastic gradient descent with a learning rate of 0.0001 and a momentum of 0.9. Categorical cross-entropy was used to classify the results of the CNN into the diagnosis labels. Training was done on an Nvidia Titan Z GPU and took approximately 20 epochs to complete each experiment. Depending on the dataset size, epoch training times ranged between approximately 45 minutes and 1.5 hours.

2.6. Results and Discussion

Table 2.2 details the results of the classification experiments. The experiments were each performed 5 times holding out a different random subset of the data for validation. The mean age and gender splits for each validation subset are shown in Table 2.3. One can see that each validation subset is not biased by patient age or gender. The networks in each experiment were trained independently and from scratch using different random weight initializations. The mean validation accuracies in percentages are reported along with the corresponding standard deviations. To reiterate, the structure of the MRI and PET networks are identical, as they both take in a single volume and have the same number of trainable weights. The fusion network takes in two volumes, one from each modality, into parallel branches that each have half the number of weights as a single MRI or PET network. Aside from a few extra weights at the end of the fusion network, the total number of weights in all three networks is roughly the same. Additionally, the fusion network used the same number of data points as the PET network, but each included two volumes instead of one. The MRI network was able to use more data points due to the larger number of MRI scanning sessions. Consequently,

two MRI experiments were run: one using all available MRI data and one with a limited dataset of the same size as the PET dataset.

Trial	MRI (Full)	MRI (Limited)	AV-45 PET	Fusion
1	91.59	78.38	84.77	94.45
2	90.46	80.43	90.65	93.47
3	84.45	66.67	80.98	90.30
4	84.64	69.97	84.92	93.41
5	86.29	73.33	84.45	90.15
Total	**87.49 ± 3.33**	**73.76 ± 5.72**	**85.15 ± 3.48**	**92.34 ± 1.95**

Table 2.2. MRI and Amyloid PET Fusion Classification Accuracies (%)

Trial	Age	Gender Split % (M/F)
1	74.98 ± 7.30	46.15/53.85
2	74.60 ± 6.72	44.57/55.43
3	74.29 ± 6.91	51.65/48.35
4	75.83 ± 7.72	57.87/42.13
5	73.79 ± 7.64	54.35/45.65
Total	**74.70 ± 7.30**	**51.53/48.47**

Table 2.3. Classification Subject Age and Gender Breakdown

To begin, the full data MRI network is able to classify with 87% accuracy. While this number is respectable, the performance could improve beyond 95% by employing techniques such as those described in [**29, 71, 34**]. However, we once again underscore that the goal is to compare the performance of the data modalities in the most balanced way possible. The inclusion of some of the more specific techniques in [**29, 71, 34**], such as pre-training the CNN filters with an autoencoder, does not enhance the modality comparison. Rather, the added complexity may obfuscate the findings if the pre-training effectiveness differed. That said, the full data MRI results do not tell the full story in the context of modality fusion. Because the MRI dataset is much larger than that of the PET, the potential for the network to learn is greatly increased. Thus, a direct comparison between the full data MRI

network and the PET network could be misleading, as the MRI results may be inflated. Thus, one must look at the limited data MRI classification results when comparing the modalities and fusion head to head. In this case, because the dataset was limited to less than half of the available scans, the network was only able to achieve an accuracy of 74%. This discrepancy is somewhat expected, but moreover it highlights a large point about the availability of training data. Given this accuracy differential for the MRI data, one can imagine the potential benefits to the PET and fusion results as the number of available amyloid PET scans increases. On that note, it can be seen that the PET network performs much better than the MRI network trained with the equivalent data size. The accuracy of 85% is even comparable to the full data MRI network, despite being trained with far fewer examples.

To properly discern the distinction between the MRI and PET performance, one must examine the biological facets of the modality. Amyloid accumulation has been hypothesized to begin more than two decades before symptoms occur [**39**]. In a longitudinal study of dominantly inherited Alzheimer's disease [**27**], elevated amyloid PET signals were found 22 years before expected onset of symptoms.

Separate from the CNN pipeline, a standard method, previously described [**50**], was used to calculate the total amyloid burden. Briefly, FreeSurfer [**19, 20**] was used to parcellate the T1-weighted MRI scan taken closest to the amyloid PET visit. Whole cerebellar referenced cortical regions normalized by volume were used to calculate a single weighted standard uptake value ratio (SUVR). The previously defined cutoff of ≥ 1.11 was used to define amyloid positivity [**50**].

In the first set of classification experiments, out of the 11 amyloid PET scans that were incorrectly classified, 7 were controls and 4 were Alzheimer's dementia cases. All 7 control

cases had elevated amyloid SUVR $\geq$ 1.11 (average SUVR 1.42 $\pm$ 0.12). Two Alzheimer's dementia cases were amyloid positive (i.e., true misclassification) and two Alzheimer's cases were amyloid negative (average SUVR 0.95 $\pm$ 0.03) and therefore are unlikely to have underlying Alzheimer's disease neuropathology. If the 7 elevated amyloid controls and 2 amyloid negative AD cases are removed, then the effective PET classification accuracy rises from 85% to 97%.

The newly proposed NIA-AA research criteria for Alzheimer's disease [**37**] points out that amnestic dementia diagnoses are not sensitive or specific for AD neuropathologic change. From 10 to 30% of individuals classified as AD dementia do not display AD neuropathology at autopsy [**68**] and 30 to 40% of individuals classified as unimpaired healthy have AD neuropathologic change at post-mortem examination [**7, 72**]. The proposed CNN here is capturing this mismatch between biomarker and diagnosis. The CNN labels healthy individuals with high amyloid PET as AD and those with Alzheimer's dementia and low amyloid PET as non-AD. Thus, while the phenomenon negatively impacts performance in this context, amyloid PET scans may be adept in a longitudinal study because elevated amyloid precedes symptom onset.

With this in mind, a few points regarding the comparison between MRI and amyloid PET can be stated. First, it is clear that the network benefited from the use of the full training set. Therefore, one can expect the PET performance to increase as well once amyloid scans become more readily available. This potential improvement may not be on the same scale, given that the PET performance is already higher than the MRI performance using the same training set size. This PET performance is likely due to the fact that amyloid accumulation may occur far ahead of symptom onset, which in turn may occur in advance of structural changes that would be detectable with an MRI. Moreover, the false positive cases of the

PET network all had elevated amyloid levels. This indicates that the network is effective at deducing elevation of amyloid levels from the PET scan and converting this information into a disease status determination. Furthermore, in these false positive cases, it is quite possible that these patients develop Alzheimer's neuropathology at a later time. This in turn would support the justification for using amyloid PET in a longitudinal prediction case rather than structual MRI data alone.

The final noteworthy result of the investigation is that the fusion network outperformed both the individual MRI and PET networks. Additionally, the fusion network outperformed the full data MRI network despite the fact that less data points were used. Again, having more PET scans available in the fusion case may further improve the accuracy. The fusion performance is consistent with the other results [**101, 93, 87**], despite the fact that these investigations use FDG PET rather than amyloid. One can see back in Table 2.1 that the MRI and FDG PET classification accuracies are rather comparable in all cases, while the fusion results are greater than either individual modality. In our case, the amyloid PET results are much better than the MRI results when using the same amount of training data, and the fusion provides a similar benefit to accuracy. That said, one cannot make a direct head to head comparison between amyloid PET and FDG PET from this investigation alone due to the fact that different biological markers, data subsets, pre-processing methods, and classification algorithms were used. A further investigation that holds these factors constant would be required. Nonetheless, this investigation still clearly demonstrates the discriminative power of the amyloid PET modality and the potential for even further gains when fused with MRI.

2.7. Conclusion

This work compared the effectiveness of the T1-weighted MRI and AV-45 amyloid PET modalities in the context of computer aided diagnosis using deep neural networks. Specifically, two identically structured CNNs were designed and trained on MRI and amyloid PET data that were pre-processed to be as fairly compared as possible. The classification results indicate that MRI data is less conducive to neural network training than amyloid PET data to predict clinical diagnosis. However, a network that uses both modalities, even with the same number of trainable weights, will achieve higher accuracy. This indicates that the two data types have complementary information that can be leveraged in these kinds of tasks. This phenomenon was also placed into the biological context of amyloid vs. MRI.

While these results are a step forward in the optimization of computer aided diagnosis tools for AD, the value from this investigation must be utilized in further applications. A natural extension can be made to looking at AD patients on a functional spectrum rather than distinct diagnosis categories. Additionally, as previously alluded to, longitudinal studies that use several scanning sessions of multiple modalities may not only improve classification performance, but also allow the ability to perform more complex tasks such as predicting future cognitive decline irrespective of clinical phenotype. Given the value these results would provide to clinicians, we investigate regression and longitudinal prediction in the following chapter.

CHAPTER 3

Alzheimer's Disease Cognitive Score Regression and Longitudinal Prediction using Convolutional Neural Networks

3.1. Chapter Abstract

Deep learning methods have had great effect in the automated classification of Alzheimer's disease status from neurological scans. However, the classification paradigm is limited in its functionality due to the coarse quantization of categorical labels. As such, we propose neural network architectures that change the paradigm from classification to cognitive score regression and longitudinal prediction. Not only does this allow for more nuanced delineation between outputs, it also has more utility in the healthcare space. We demonstrate that these networks can perform regression and longitudinal prediction within the margin of uncertainty inherent in the Mini Mental State Examination (MMSE), a widespread cognitive measurement used in the clinical assessment of Alzheimer's disease.

3.2. Introduction

Alzheimer's disease (AD) is a neurodegenerative disorder that may affect as many as 10 million people by the year 2025 [**31**]. Consequently, there is a great need for tools that can automatically asses disease status and predict cognitive abilities. Often, this is formulated as a classification problem, where an algorithm outputs a binary disease label indicating the presence of the disease or lack thereof. In some cases, this disease label is further quantized to include transitional states, such as the mild cognitive impairment (MCI) state in the

progression of Alzheimer's disease. Nonetheless, this coarse quantization may not be the optimal delineation in this domain; rather, a method for predicting cognitive function on a continuous spectrum is a more clinical useful tool. In addition to providing a more sensitive metric for determining disease status, the algorithm would not require clinical judgements during the training phase. As such, this setup would provide more robust predictions for patients on the border between disease states (e.g. healthy and MCI, or MCI and AD). In addition, the ability to predict this cognitive outcome at a future time would greatly increase the efficacy of clinical intervention. This longitudinal prediction is fundamentally more practical and relevant to the field than the standard classification paradigm.

This work is a direct extension of our former investigation into the predictive power of neuroimaging modalities in Alzheimer's classification from Chapter 2. There, we used a 3D convolutional neural network (CNN) trained on either T1-weighted MRI or AV-45 amyloid PET scans to perform binary classification (healthy vs. AD). Afterwards, we trained a network that consisted of two parallel network branches, one for MRI and one for PET, and fused the resulting features before yielding a prediction. Modality comparison will be a major component of this investigation as well, but performed in the regression space as opposed to the classification space. Furthermore, the imaging modalities will be fused with non-imaging modalities, namely patient age and relevant genetic information. Lastly, the architecture was modified for longitudinal prediction and analogous modality comparisons were performed. The following sections will present an overview of related approaches, the pre-processing and network architectures, implementation details, and a discussion of the regression and longitudinal results.

3.3. Related Work

The techniques and methodologies utilized in automated Alzheimer's diagnosis have ranged in complexity and efficacy. Additionally, much of the work tackles the classification problem, rather than the cognitive score regression problem. Initial methods often used linear classifiers or support vector machines (SVM), such as an SVM that classifies gray matter tissue maps [**46**] or spherical harmonic coefficients from the hippocampus [**23**]. In [**57**], a template selection and clustering procedure preceeded the use of gray matter density maps in an SVM. A comparison study between SVMs and other linear classifiers using an aggregate of Alzheimer's datasets was performed in [**78**]. The authors in [**118**] use a kernel-based approach to combine features extracted from both MRI and FDG PET images. Of the linear approaches, the work in [**116**] is most closely related to our current investigation. The authors perform simultaneous classification and cognitive score regression with a support vector model that uses MRI, PET, and cerebrospinal fluid (CSF) images.

Eventually, the prevalence of linear methods declined and non-linear approaches began to dominate the algorithmic landscape. That said, there is still much heterogeneity within the class of non-linear algorithms. Deep Boltzmann machines (DBM) were used to extract features from MRI and FDG PET images [**94, 93**] as well as CSF and cognitive testing data [**53**], but the classification in all these cases was still performed with an SVM. The authors in [**106**] employed a fully-connected network that was aided by spatial neighborhood regularization. Yet, due to the visual nature of neuroimaging data, the CNN has become one of the prevailing tools in this domain. Traditional CNN architectures like LeNet-5 [**52**] and GoogleNet [**95**] were modified to operate on fMRI data ([**81**] and [**82**], respectively). The investigation in [**54**] developed a multimodal classification CNN that also accounted for

missing one of the two imaging modalities (MRI or FDG PET). Convolutional autoencoders [**33**] have also been used, such as in the case of [**58**], where a stacked autoencoder was used to extract MRI and FDG PET features. 3D MRI images were passed through an autoencoder to pre-train a CNN [**34**], and this was further developed in [**101**] to include FDG PET. Autoencoders were also used on 2D MRI slices, as in [**29**]. The authors in [**71**] implemented both 2D and 3D variations of this procedure and compared the performance.

Work tackling the Alzheimer's classification problem is abundant; however, investigations into cognitive score regression are somewhat less popular. The most relevant to this work is [**9**], in which the authors perform cognitive score regression on the Mini Mental State Examination (MMSE) and Alzheimer's Disease Assessment Scale (ADAS-13) using structural MRI. There are also some preliminary longitudinal prediction results. Another related publication [**112**] looked at these exams in a multimodal fashion by inputting MRI, FDG PET, and CSF images. The authors in [**109**] perform similar experiments, but use a graph-based methodology to remove the assumption of structured data, while the authors in [**102**] use Bayesian analysis. Current methods in longitudinal prediction are also rather sparse. Some non-imaging methods include [**115**] and [**97**]. The former uses a Bayesian model on genetic data to predict MMSE changes for each classification label separately, while the latter uses a neural network on some demographic features like education as well as handcrafted features like a categorized rate of disease progress. Some studies look at predicting patient conversion from one disease status to another, such as in [**64**]. That said, the only deep learning-based approach to longitudinal MMSE prediction that uses neuroimaging modalities is [**96**]. This work handles the temporal nature of the data in a similar manner to ours, but the treatment of features is rather different. In [**96**], the authors extract features from the imaging (and non-imaging) modalities and then perform a selection process before sending them as input

to the network. In our case, we will use a trainable convolution approach to extract features and perform the prediction. A summary of these works is shown in Table 3.1.

Method	Task	Input Modalities	Reported Results
[**9**]	Regression	MRI structural features	r = 0.55
[**112**]	Regression	ROI features from MRI, FDG PET, CSF	r = 0.697
[**109**]	Regression	ROI features from MRI, FDG PET, CSF	r = 0.745
[**102**]	Regression	ROI volumes from MRI, FDG PET	r = 0.735
[**115**]	Longitudinal Prediction	Demographics, genetic info, test scores	r = 0.632
[**96**]	Longitudinal Prediction	MRI, amyloid PET, CSF, and non-imaging	MSE = 4.77

Table 3.1. Related works performance; results self-reported as Pearson correlation coefficients (r) or mean squared error (MSE)

These publications predominantly use correlation coefficients as a metric for regression success, and often focus on joint classification accuracies. However, this work will exclusively report regression results because our classification results were reported in the previous chapter. Furthermore, the vast majority of fusion studies look at the conjunction of MRI and FDG PET. Typically FDG PET, which measures metabolic activity of glucose, is used but only provides incremental information when combined with MRI. Our work will combine MRI and amyloid PET. This distinction is important, as amyloid PET measures beta amyloid, which has been identified as a strong predictor of Alzheimer's disease [**39**]. The work in[**96**] was the only one to also use amyloid PET, but once again the authors extracted handcrafted features from the amyloid PET image as opposed to learning the features in a CNN-based fashion. Another distinction of our work is that our pre-processing pipeline, originally outlined in Chapter 2, is advanced in its handling of both modalities and preparation for longitudinal prediction. Fundamentally, the cognitive score regression investigation is nascent in its choice of modality, preparation of data, and reporting of results in the

context of clinical diagnoses. The longitudinal prediction investigation is also unique in its preparation of data and for its handling of features and fusion methodology.

3.4. Methodology

Methods for automated cognitive score regression have many of the same design considerations as automated diagnosis algorithms. One may, for example, decide to use neuroimaging features at the individual voxel level or at the region level. Alternatively, one could pass the image directly into the model, but a decision between 2D slices and 3D volumetric inputs must still be made. One comparison study [**14**] observed that whole brain models tended to perform better than models that take region-based features as input. Furthermore, pre-processing techniques such as affine registration can significantly affect performance. These findings, in addition to previous experimentation, led us to design a pipeline with an advanced pre-processing scheme and a network that takes full 3D volumes as input. The following subsections will summarize our pre-processing pipeline, originally described in the previous chapter, and the neural network redesign to tackle the regression and longitudinal prediction problems.

3.4.1. Pre-Processing

Our pre-processing pipeline was outlined extensively in Chapter 2, so here we will present a summation of the pipeline mechanics and design rationale. A diagram of the pipeline can be seen in Fig. 2.1. Each subject in the dataset has a potentially different number of MRI and amyloid PET scans, so the pipeline is run separately for each individual in the dataset.

To begin, N4 bias field correction [**99**] is applied to each of the MRI scans. This algorithm is a standard in removing any potential low frequency bias components in the image that

can result from the scanner. Next, each of the MRI scans are registered to the first time point scan using a linear affine transformation. For example, if a patient has MRI scans at baseline, 6 months, and 12 months, the 6 month and 12 month scans are registered to the baseline scans. At this point, all of the scans are geometrically within the baseline space. These are averaged to create a template to which all of the scans (including the initial time point scan) are registered. This accounts for any temporal discrepancies, which is valuable for our longitudinal studies. Afterwards, the scans are registered to the standard MNI152 brain atlas [**21**]. This is common in many neuroimaging pipelines, as it accounts for any discrepancies between individuals in the dataset. Finally, extraneous information is removed from the scans in a skull stripping process performed by simply removing non-brain regions in the MNI152 brain mask.

At this point, all of the MRI scans are sufficiently pre-processed. The intermediate templates created in this first part of the procedure are then used in the PET portion of the pipeline. Because the first 10 minutes of the amyloid AV-45 PET scans were reconstructed into two 5 minute frames, the two frames are motion corrected by co-registering them together. Then, the scans are registered to the MRI baseline average and MNI152 template using the same warping method as before. Skull stripping is also performed in the same fashion, and the resulting PET scans are in the same space as the corresponding MRI scans. Thus, all of the neuroimaging scans are accounted for in terms of longitudinal differences within each subject and differences between subjects in the database.

3.4.2. Regression Networks

The base neural network architecture used in these experiments is a modification from the classification CNN used in Chapter 2. While the majority of the feature extraction portion

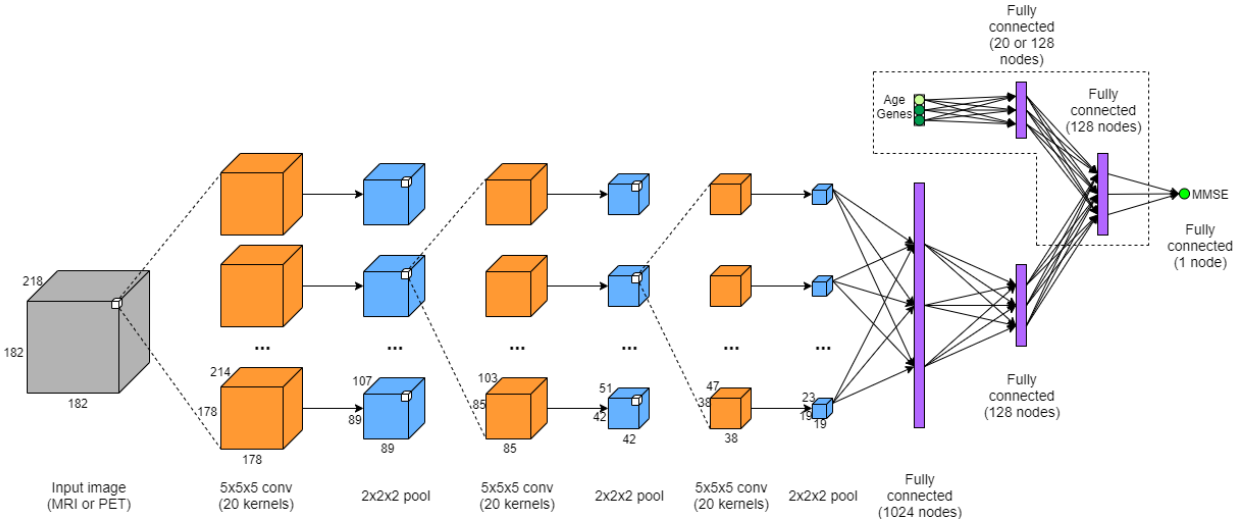


Figure 3.1. Neural network architecture for single neuroimaging modality regression experiments

of the architecture remained the same, the latter part of the network was modified to return a single output corresponding to the MMSE score (as opposed to the 2-length vector for binary classification) and to incorporate the addition of some non-imaging features. Fig. 3.1 shows the regression network for a single MRI or PET scan as input. The input image is passed through a series of alternating 3D convolutional layers with 5x5x5 kernels and max pooling layers of size 2 in each dimension, in the same fashion as before. The ReLU activation function follows each convolutional layer. After three such layer pairs, the now 20 23x19x19 features are reshaped the sent through a fully connected layer of size 1024 and subsequent fully connected layer of size 128. These layers also employ ReLU activations.

At this stage, one of two things can occur depending on the inclusion on non-imaging data in the experimental setup. In the case where no non-imaging data is used (i.e. no age and genetic data), the 128-dimensional feature vector is fully connected to a single node at the end of the network. A sigmoid activation function is applied on this node to yield the output corresponding of a patient's MMSE score. However, the network was also tested in

settings in which non-imaging information, namely patient age and APOE4 gene expression, was used. In this case, the non-imaging input information is concatenated into a single input vector before being passed through a fully connected layer. This was done is order to increase the dimensionality of the non-imaging features before fusion with the features from the convolutional branch. Without this, the non-imaging features of size 3 would be directly concatenated with the size 128 vector from the convolutional branch. Even if another fully connected layer was applied after this, we found that there was no difference in performance. With a fully connected layer of size 20 or 128 in between, we were able to test whether fusing features of equal size or smaller would be beneficial. Thus, after sending the non-imaging input through the first fully connected layer, the features are joined with the convolutional branch features with a fully connected layer (also of size 128 to maintain the same size) before being sent to the output. Fig. 3.1 shows this final phase of the network in the dotted lines.

Fig. 3.2 shows how the PET modality is added to this setup. In essence, two imaging network branches are created that are architecturally the same as in the previous single imaging modality network. That is, the MRI or PET volume is passed through a series of convolutional and pooling layers followed by some fully connected layers until a feature vector of size 128 is formed. However, each branch has half the number of convolutional kernels as in the single imaging modality case. This is to attempt to maintain a similar number of trainable weights between the experiments, as was done in Chapter 2. After the two neuroimaging branch features are calculated, they are fused with a size 128 fully connected layer. At this stage, these features can be combined with the non-imaging features identically as before, given that the vectors sizes are the same in both setups. With these two networks, we are able to investigate the capacity for the network to predict MMSE based

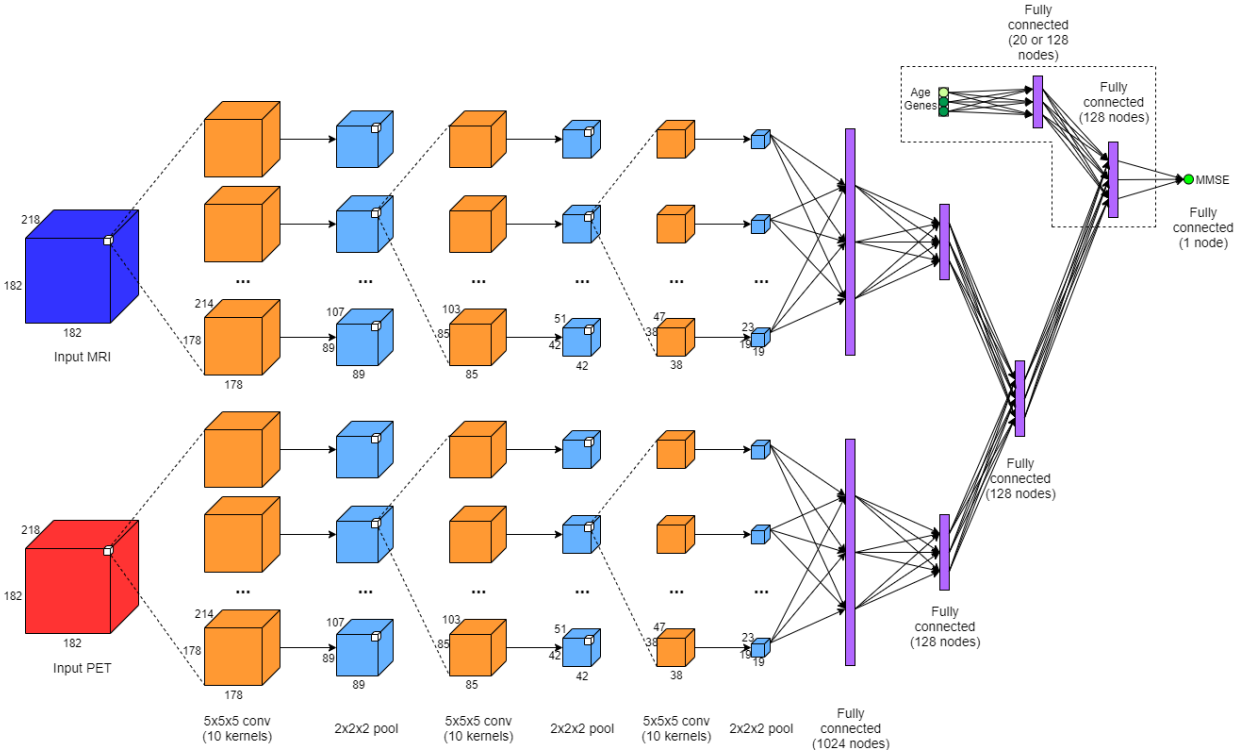


Figure 3.2. Neural network architecture for double neuroimaging modality regression experiments

on either one or two neuroimaging modalities, along with the optional presence of age and genetic information.

3.4.3. Longitudinal Networks

Figs. 3.3 and 3.4 show the modifications made to the regression networks in order to perform longitudinal prediction of MMSE. In Fig. 3.3, one can see the modifications made to perform longitudinal prediction with MRI input data. These networks take two temporally consecutive MRI scans from the same patient and predict MMSE at the third time point. In order to do so, the two MRI scans are first passed through a time distributed CNN block. This block can be conceptualized as a single CNN branch, identical to the aforementioned

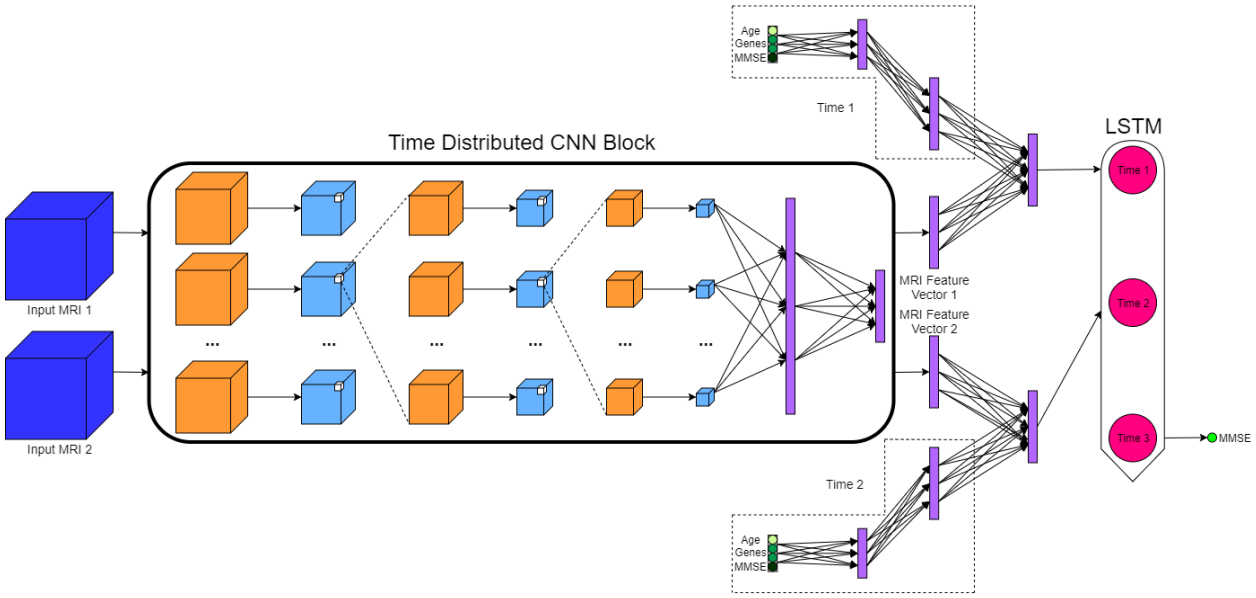


Figure 3.3. Neural network architecture for single neuroimaging modality longitudinal experiments

regression networks, that accepts separate MRI volume inputs and returns corresponding separate MRI feature vectors. This is visually displayed in Fig. 3.3, in which the time distributed CNN block is used to compute feature vectors from MRI scans at two different time points. In the double neuroimaging modality regression network from Fig. 3.2, there are two kinds of blocks. The MRI CNN block is time distributed, but the PET CNN block is not. Because multiple PET scans are not used at once, the block simply takes in one PET scan and outputs the corresponding feature vector.

After the two MRI feature vectors are found, they are fused with the non-imaging information as before. The only addition here is the presence of the previous time instances MMSE scores. Given that we are trying to predict MMSE at time point 3, it is logical that we would have access to the scores from time points 1 and 2. As such, the MMSE scores at these times are concatenated with current patient age and gene expression for the appropriate MRI data. After fusing with the MRI features, the resulting two vectors are

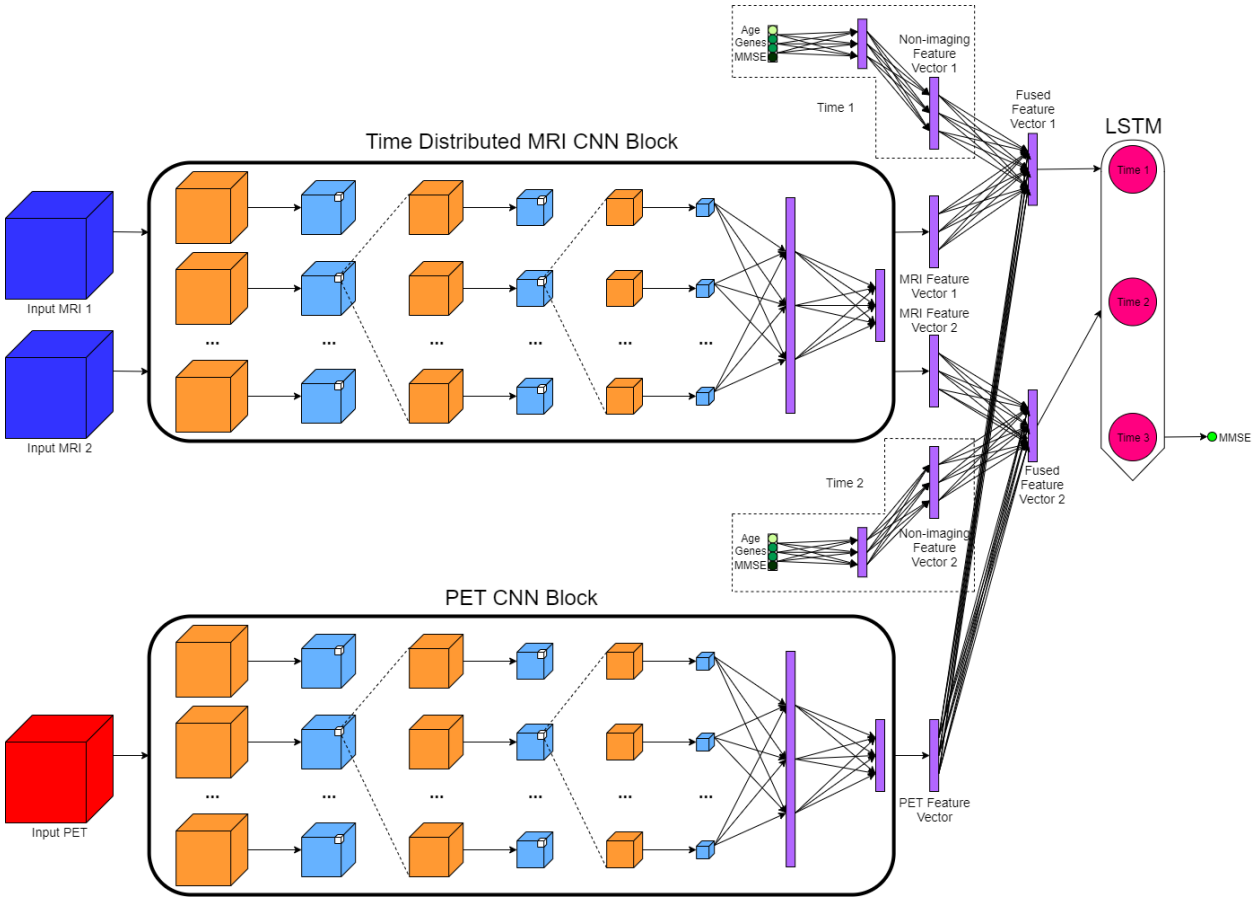


Figure 3.4. Neural network architecture for double neuroimaging modality longitudinal experiments

fed into an LSTM. LSTMs are a form of recurrent neural networks (RNN) that have shown great efficacy in time series analysis. RNNs have an internal "state" that encode information from previous time instances and passes it forward. An output at a later time step makes use of information from all previous time steps. However, a well known short-term memory problem exists in RNNs wherein vanishing gradients during training cause the network to "forget" information from further and further time steps. Thus, a long short-term memory (LSTM) cell was developed in order to combat this problem. The LSTM cells have a series of gated operations [**24**] that enable the cell to more accurately retain past information.

As a result, LSTMs have become the natural choice in temporal signal processing. In our case, we use a single layer LSTM that take in input from two time instances and predicts patient MMSE score at the third. The final output of the LSTM has the traditional sigmoid activation in order to yield the MMSE prediction.

As one would expect, we also wanted to perform neuroimaging modality fusion and include the PET scan in the longitudinal prediction. However, due to the limited number of PET scans in comparison to MRI scans in the dataset, it would not have been possible to pair MRI and PET scans at all time points. Rather, we would only be able to supply one PET scan per patient. Thus, the PET information was included in a similar fashion to the double neuroimaging modality regression network, whereby a separate CNN branch took in the PET volume and outputted a vector that was fused with the other features in a fully connected layer. Ergo, the MRI features from time point 1 were fused with the PET features, and the MRI features from time point 2 were separately fused with the same PET features. At this stage, the fused feature vectors are in the same size the the former network, and the LSTM can be used as before. The final architecture can be seen in Fig. 3.4. Along with the other networks, MMSE regression and longitudinal prediction was performed in accordance with the data specifications outlined in the next section.

3.5. Experimental Design

The regression and longitudinal prediction experiments were performed using the the Alzheimer's Disease Neuroimaging Initiative (ADNI) [**38**] dataset. The database contains a myriad of data types ranging from neuroimaging modalities to medical history and genetic sequencing. The primary data types in our investigation were T1-weighted MRI and AV-45 amyloid PET scans. As discussed, these modalities are the predominant input into our

network architectures. Because fusion is a major component of the current and previous investigations, only data from subjects with at least one MRI and one amyloid PET scan were used in the regression experiments. An even stronger criteria was used in the longitudinal experiments, where a subject was required to have at least 3 MRI scanning sessions. This constraint could not be applied to the PET data and still retain a large enough population, so only one PET scan per subject was used in the longitudinal fusion case. Patient age and genetic information in the form of two APOE4 alleles were used as non-imaging inputs in the regression experiments. MMSE scores are used as target outputs, and previous time instance MMSE scores are used in the longitudinal prediction case. We had a temporal MMSE constraint in our subject selection for the experiments, in that we only used scans that had a corresponding MMSE score recorded within two months of the neurological scan date. This was to ensure that image-MMSE pairs could be formed in a fair manner.

These subject selection criteria resulted in data from 630 subjects being used for the MRI regression experiments, with 1654 MRI scans in total. The relatively limited quantity of PET data resulted in 488 scans from 382 subjects in the PET regression experiments. Because the PET data was the limiting factor, the fusion regression experiments had the same data size as the PET experiments, with only a selection of MRI scans being used in the process. The longitudinal experiments naturally had a smaller sample size, given that each input-output pair required three consecutive MRI scanning sessions. 492 MRI triplets (2 input scanning sessions, 1 output scanning session) from 318 patients were used in the longitudinal prediction experiments. A single PET scan from each subject was fused with the MRI features in this case rather than forming PET triplets, again due to data availability. The main categories of experiments were as follows: MRI regression, PET regression, fusion regression, MRI longtidinal prediction, and fusion prediction. Within these categories, some

additional trials were run with different hyperparameter settings, such as which non-imaging modalities were used and size of the corresponding feature vector.

The dataset in each experiment was divided into training and testing components using an 80-20 split. While these splits were performed randomly in order to generate multiple training folds, another consideration was made to ensure that no single patient's scans were in both a training and testing subset. Without this additional step, the networks have the potential to overfit to a patient's identity as opposed to identifying patterns caused by the underlying disease. Other works in this domain may have fallen into this trap and consequently could have artificially inflated results. Each experiment was performed 5 times with a different training-testing fold, and the networks in each case were trained from scratch with randomly initialized weights. We report the mean absolute error in predicted MMSE scores for both the regression and longitudinal experiments.

The networks themselves were constructed in Python using the Tensorflow [**1**] framework as a base implementation. Stochastic gradient descent with a learning rate of 0.0001 and momentum of 0.9 was used as the optimizer. The loss function in both the regression and longitudinal experiments was mean squared error between the target and prediction MMSE scores. Training was performed on an NVIDIA Titan V GPU and took approximately 250 epochs for the regression experiments and 1000 epochs for the longitudinal experiments.

3.6. Results and Discussion

The results from the imaging-only regression experiments are shown in Table 3.2. In this case, only the neuroimaging modalities are used to predict a patient's MMSE score at a given time. The MRI and PET columns of this table show the results when the architecture from Fig. 3.1 is used; that is, a single volume is passed through a single CNN branch. However,

Trial	MRI	Amyloid PET	Fusion
1	2.109	1.550	2.262
2	2.137	1.976	1.822
3	1.779	2.153	2.002
4	2.079	1.982	2.108
5	2.023	1.885	1.890
Total	**2.025 ± 0.1440**	**1.909 ± 0.2229**	**2.0168 ± 0.1751**

Table 3.2. Regression experiments, no non-imaging information (MMSE mean absolute errors, MMSE scores range between 0 and 30)

because the age and genetic information are not yet used in this case, the non-imaging branch (within the dotted lines in Fig. 3.1) is not used. Therefore, the features after the second fully connected layer in the CNN branch are used to directly compute the predicted MMSE score at the output.

One can see the mean absolute MMSE errors for the 5 folds/trials in each modality setup, as well as the corresponding average error. This metric was chosen primarily due to its interpretability in a clinical context. Unlike mean squared error or root mean squared error, mean absolute error (MAE) corresponds to a direct uncertainty in MMSE score measurement, which we will also shortly discuss with relation to the uncertainty inherent in the MMSE exam. To begin, one can see that the network is able to predict a patient's MMSE score with MAE of 2.025 points when training and testing on MRI volumes. This is in contrast to the case when amyloid PET is used, where the network has a MAE of 1.909 points. On the one hand, this relative improvement in performance with the PET modality is in line with our previous findings that amyloid PET has more discriminative power than MRI in the classification setting. However, the relative gain in performance is much smaller and within the reported standard deviations. This is further evidenced by the lack of improvement in the fusion scenario, where both MRI and PET are used as inputs. In the classification case,

Trial	MRI	Amyloid PET	Fusion
1	2.116	2.037	2.328
2	1.985	1.872	1.843
3	1.831	2.063	1.968
4	2.225	1.668	1.691
5	1.740	1.717	1.926
Total	**1.979 ± 0.1990**	**1.871 ± 0.1798**	**1.951 ± 0.2357**

Table 3.3. Regression experiments, age and genetic information fused via 20-dimensional vector (MMSE mean absolute errors, MMSE scores range between 0 and 30)

the fusion scenario had a marked 7% improvement in classification accuracy over the PET-only system; but, our results here indicate that fusion had no significant impact on MMSE regression performance.

While this may seem like a stark departure from previous intuition, we believe this can be explained by the uncertainty inherent in the MMSE itself. A study looking at the variability in MMSE scores in patients with probable Alzheimer's disease found the standard deviation of measurement error in the exam to be 2.8 points [**13**]. This suggests that there is a lower bound to achievable MMSE MAE. We do not believe that this bound is necessarily at the 2.8 level, as our results are consistently below this threshold; yet, the fact that our results for all modalities are well within this uncertainty implies that our methods are achieving acceptable results given the experimental setup. Thus, we believe that the performance differentiation between the MRI, PET, and fusion setups is not pronounced due to the fact that all the modalities are sufficiently able to predict MMSE scores within the inherent level of uncertainty in the exam itself.

A similar story can be seen in Tables 3.3 and 3.4, which show the results of MMSE regression when non-imaging information is fused with the existing imaging modalities. Both instances use a vector of patient age and APOE genetic information as input, but the size of

Trial	MRI	Amyloid PET	Fusion
1	2.063	2.000	2.163
2	2.022	1.862	1.854
3	1.811	2.075	1.928
4	2.205	1.684	1.665
5	1.806	1.801	1.937
Total	**1.981 ± 0.1718**	**1.884 ± 0.1560**	**1.909 ± 0.1790**

Table 3.4. Regression experiments, age and genetic information fused via 128-dimensional vector (MMSE mean absolute errors, MMSE scores range between 0 and 30)

the fully connected layer after the input layer is different. Table 3.3 refers to the case where the layer is of size 20, while Table 3.4 shows the results when the layer is of size 128. The latter experiment is a case where the non-imaging feature vector is the same size as the imaging vector before fusion. The intent of this set of experiments was to determine whether the dimensionality of the non-imaging features would be a major factor in performance. The key problem was the discrepancy between the 128-dimensional neuroimaging feature vector at the end of the convolutional branch of the network and the small 3-dimensional non-imaging features. A direct concatenation of these two vector would not necessarily provide adequate weighting to the non-imaging features. Thus, the intermediary fully connected layer acts as something of an up-sampling measure, akin to how some autoencoder networks use layers larger than the dimensionality of the latent space to "decode" the features. Therefore, the 128-dimensional vector attempts to treat the non-imaging features with equal weight to the imaging features, while the 20-dimensional vector aims to upsample the non-imaging features more conservatively.

One can see a more detailed breakdown of performance in Fig. 3.5, which shows the predicted vs. ground truth MMSE for one specific trial in a regression experiment. Each point represents a single patient's MMSE prediction in the test set. A red line of slope one

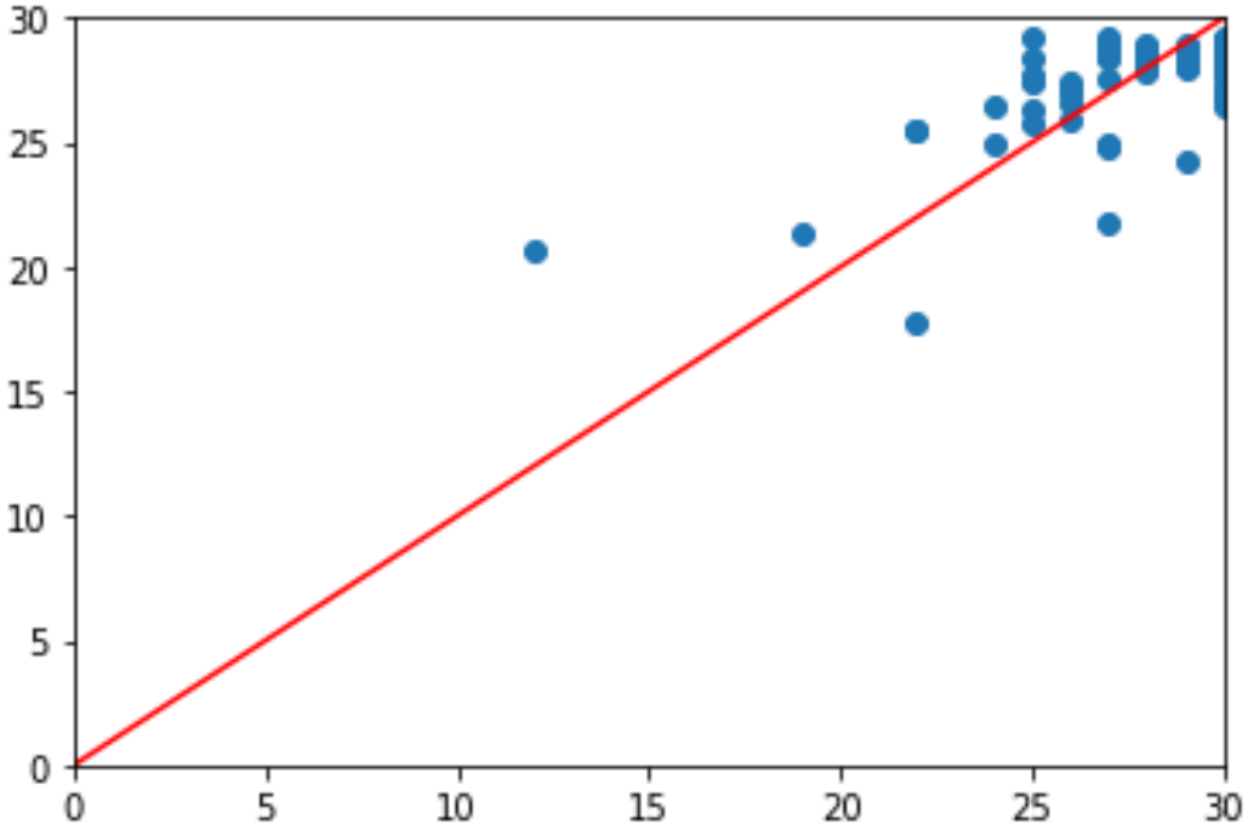


Figure 3.5. Predicted vs. True MMSE (regression, fusion, 128-dimensional non-imaging vector)

is shown to indicate ideal performance. While the errors are generally distributed around this trendline, a few observations can be noted. For one, directly from the data, one can see that the distribution of true MMSE scores is skewed towards scores ¿ 25. This is quite logical, as many individuals in the dataset are healthy, or perhaps only exhibit mild cognitive decline. This ties into another observation that can be made regarding the directionality of MMSE errors: generally, the model overshoots true MMSE. This is often only by one to two points, but there can be larger errors, such as the one shown in the far left of the plot. The true MMSE is approximately 12, but the predicted value is over 20. This may be explained by the fact that the training data likely does not contain many examples with

Trial	MRI	MRI + non-imaging	MRI + PET + non-imaging
1	2.208	2.288	3.001
2	1.717	1.649	2.228
3	1.776	1.858	2.192
4	1.929	1.881	2.240
5	1.673	1.756	2.236
Total	**1.861 ± 0.2168**	**1.886 ± .2426**	**2.379 ± 0.348**

Table 3.5. Longitudinal experiments, age and genetic information fused via 20-dimensional vector (MMSE mean absolute errors, MMSE scores range between 0 and 30)

such low MMSE scores, but another contributory factor may be linked with the skewed dataset toward healthy and low MMSE individuals. That said, contrary examples exist, as one can see with the patient whose true score was approximately 22 and predicted score was approximately 18. Thus, while there is some credence to suggest that the model is generally overshooting MMSE, this is not always the case.

All this said, while there is some potential benefit to adding the non-imaging features, the benefit is too small to make any concrete claims. It is logical that additional information should benefit the model, as disease progression is naturally accelerated with aging and the APOE genes are correlated with the presence of Alzheimer's. However, we again run into the MMSE uncertainty problem. Given that all of the results in both sets of experiments fall within the MMSE uncertainty of 2.8 points, it is not possible to tease out any distinctions between modalities or feature vector sizes. As a result, we reiterate our previous claim, now in the context of non-imaging information: all of the experimental setups are sufficiently able to perform MMSE regression within a reasonable amount of error, but improvements between modalities and among the non-imaging feature vector sizes cannot be determined given the inherent MMSE uncertainty.

Finally, Table 3.5 shows the results of the longitudinal predictions of MMSE. Unlike our previous sets of classification experiments, we are unable to show results for MRI, PET, and fusion in all cases. Recall that the limited number of longitudinal PET scans prohibits our ability to perform direct modality comparisons. Instead, we show longitudinal predictions results using MRI input alone, in conjunction with non-imaging features, and also using both non-imaging features as well as a single PET scan as reference. The results here indicate that the MRI input is able to predict future time point MMSE with an average error of 1.861 points, which is actually slightly less than in the previous regression cases. While this may be surprising given the increased complexity of the task, the longitudinal experiments have an advantage over the singe time regression experiments. Because the longitudinal data makes use of previous time point MMSE scores, the network is potentially only learning the residuals of output scores. That said, the MRI + non-imaging error still follows the pattern seen in the regression experiments, where the addition of non-imaging information did not provide noticeable benefits to performance.

Perhaps the most interesting result is the final set of experiments, where the MRI and non-imaging information was combined with a single PET scan as shown in Fig. 3.4. One would expect fusion to improve, or at the very least not detract from regression performance; yet, these results show a more prominent increase in error. We believe this is because the single PET scan was, by definition, only showing information from a single time point. Since this scan had to be fused with features from two different time points, it is possible that the PET features and MRI/non-imaging features could have provided conflicting information. For example, a PET scan from an earlier part of the patient's history may indicate a high MMSE value, but could be paired with MRI and non-imaging features that indicate a lower value. This conflicting information would likely negatively impact the accuracy of the classifier.

We initially hypothesized that the differences in PET scans between two time points might be small enough to allow for this method of fusion, but experimentally it is likely not the case. As such, while we made an attempt to make a fusion comparison in the longitudinal prediction studies, this particular setup was deleterious rather than beneficial. With more available PET scans, a true fusion setup could have been tested and may have provided the predicted increase in performance.

3.7. Conclusion

In this work, we extended the traditional paradigm of automated Alzheimer's disease classification to solve the cognitive regression and longitudinal prediction problems. A convolutional neural network was used to extract features from full brain MRI or PET volumes. These features were then optionally fused with non-imaging information, specifically patient age and APOE4 genetic expression, in order to predict cognitive abilities on the MMSE scale. Then, an LSTM component was added to the convolutional network in order to allow for longitudinal prediction of MMSE. It was found that the regression and longitudinal networks are able to perform their respective tasks with a mean absolute error of less than 2 MMSE points. This is within the inherent uncertainty of the MMSE exam, which was found to be 2.8 points in a previous study. However, this also means that performance distinctions between modalities and fusion scenarios were not as apparent as in the classification situations.

As such, one direction for future study would be to circumvent this MMSE uncertainty problem by using a different measure of cognitive function, such as the ADAS-Cog. Given more precise measures of cognition would undoubtedly improve the ability for the network to accurately predict such values. Additionally, as discussed in the longitudinal PET fusion

scenario, a more sophisticated fusion setup may be necessary if no additional PET data can be obtained. One could also look to optimize the architectures themselves, as the architectures here do no employ many of the more nascent developments in CNNs. For instance, perhaps residual connections could benefit the LSTM if the network is indeed only learning the residual changes in MMSE scores between time points. If any of these changes pushed the error of the longitudinal predictions lower, the benefits to clinicians and patients could be tremendous.

CHAPTER 4

Background on Feature Visualization

Despite the relative success of convolutional neural networks (CNNs), as one can see in the previous two chapters, these algorithms have always had a major drawback related to interpretability. In many situations, understanding how a network makes a decision is just as important as its ability to make the correct decision. In this vein, simpler machine learning models and linear classifiers are much easier for humans to discern. In order to increase human interpretability of CNNs, one can create visualizations that capture information about a network's function. Then, by analyzing these images in a qualitative fashion, one can develop actionable intelligence for network modification and improvement. For example, by visualizing the filters in a CNN, one can ascertain what kinds of visual elements are found by the convolutional kernels. Early intuition postured that filters in early layers of a CNN would look for edges and simple gradients in an image whereas filters in deeper layers would be tuned to locate more complex geometric structures. This was supported by some some simple visualizations of CNN feature maps. However, this idea can be extended to greater effect.

In deep visualization, one solves an optimization problem that generates images that can describe several facets of a CNN. For instance, the main deep visualization technique that will be discussed in the following chapter, activation maximization, can be used to generate the previously described filter maps. However, it additionally can be used to create representations of the output classes in a classification network. These images indicate the dominant

visual elements the network looks for in order to perform classification. By identifying these elements and comparing them to human perception, one can determine whether the machine is performing the task in the same way as a human. Furthermore, if there are discrepancies, one may glean insight into how to combat any shortcomings of the network. With such a system in place, these networks become less of a "black box" and are more easily manipulated. To this end, deep visualization is able to address the "how" question that is otherwise very elusive in the domain of CNNs. In the following chapters, we will examine the application of activation maximization to the understanding of CNN training, examination of capsule neural network features in comparison to convolutional features, and see whether activation maximization can be leveraged to improve the controllability of generative outputs.

CHAPTER 5

Visualization of Feature Evolution During Convolutional Neural Network Training

5.1. Chapter Abstract

Convolutional neural networks (CNNs) are a staple in the fields of computer vision and image processing. These networks perform visual tasks with state-of-the-art accuracy; yet, the understanding behind the success of these algorithms is still lacking. In particular, the process by which CNNs learn effective task-specific features is still unclear. This work elucidates such phenomena by applying recent deep visualization techniques during different stages of the training process. Additionally, this investigation provides visual justification to the benefits of transfer learning. The results are in line with previously discussed notions of feature specificity, and show a new facet of a particularly vexing machine learning pitfall: overfitting.

5.2. Introduction

Convolutional neural networks (CNNs) have provided state-of-the-art performance in a variety of computer vision and image processing applications [**51**]. Recent developments in hardware, namely GPUs, have caused an inundation of CNN-based methods. That said, a discrepancy exists between knowledge of how to construct such algorithms and knowledge of how these algorithms operate. One major criticism of CNNs in general refers to the treatment of the algorithm as a "black box", with the ultimate result of the training procedure shrouded

in mystery. Although the process of backpropagation used to modify filter weights has been thoroughly discussed, describing the function of these features has been less explored.

Algorithms that fall under the category of deep visualization strive to address such issues. At their core, these methods attempt to bridge the gap between human and machine perception by illustrating CNN features in a visual manner. This paradigm differs from some traditional views on CNN analysis that are primarily results oriented. It is common practice to judge the efficacy of any modifications to a network or dataset by the capacity to increase performance. Of course, this is a functionally logical approach to CNN design; however, not observing changes to the network features themselves is another example of the "black box" methodology. Such thinking may inhibit progress towards the next breakthrough in machine learning. It is the intention of deep visualization to aid in combatting the esoterica of CNNs.

5.3. Related Work

Deep visualization encompasses several approaches that have been described in the literature. This analysis will focus on a technique called activation maximization. The term was perhaps first coined in a 2009 publication in which the authors describe "qualitative interpretations of high level features" [**18**]. They produce visualizations from a deep belief network (DBN) trained on the classic MNIST digit classification dataset that confirm intuitions held about the learned representations. Since then, several authors have employed activation maximization and modified the procedure or usage. Yosinksi et al. [**108**] applied the method to a more complex classification problem and developed an accompanying software toolbox for interactive visualization. A 2015 investigation at Google described a technique that modified activation maximization with the purpose of creating art as "inceptionism" [**3**]. In [**69**], the algorithm was modified to highlight the multifaceted nature of

specific network neurons. Mahendran and Vedaldi [**60**] created a generalized algorithm to perform activation maximization as well as another deep visualization method: inversion.

Inversion produces a different kind of visualization that is primarily used to quantify the loss of information at increasingly deep network layers. In essence, the ability for a network to reconstruct an input image from features at a given layer signify the information retained in those layers. Mahendran and Vedaldi first described their inversion method in [**59**], and Dosoviskiy and Brox supply a different approach in [**16**]. Inversion is related to another type of visualization that uses a "deconvolutional" network to identify stimuli of individual feature maps [**111**]. This identification is akin to locating the receptive field of a feature, a concept also explored in [**113**].

A third class of deep visualization algorithms can be described as sensitivity or saliency maps, which illustrate the support of a particular feature in a given image. Simonyan et al. [**88**] compare this method with a form of activation maximization. In [**117**], the authors show sensitivity maps with evidence both for and against a particular class, while [**79**] develops heatmaps showing relevance or importance of image regions.

All of these methods yield complementary views of the information in neural network features. Because this analysis focuses on activation maximization, a more detailed explanation of the procedure is outlined in the next section.

5.4. Activation Maximization

The following explanation of the activation maximization method will synthesize information from [**108**] with additional description supplied by [**60**]. As previously suggested, the algorithm aims to create visual representations of CNN features, either at the convolutional

filter level or object class level. In this manner, the method can be cast as an inverse problem that is solved using an optimization approach. To begin, consider an RGB image that produces some activation when passed through a CNN. Yosinksi formulates the problem as in [**108**]:

$$x^* = \arg\max_x (a_i(x) - R_{\boldsymbol{\theta}}(x)) \tag{5.1}$$

where x^* is the final visualization, x is a candidate input image to the network, $a_i(x)$ is the activation for some particular unit i, and $R_{\boldsymbol{\theta}}(x)$ is some parameterized regularization function. In general, the unit i to be maximized can be the index of a filter or element in any layer of the network; however, in this case, the following analysis will only concern maximizing indices representing classes in the last layer of the network. The final visualization will be a synthetic RGB image of the same size as the input. One can also formulate a minimization to accomplish the same task, as Mahendran and Vedaldi do in [**60**], that is:

$$x^* = \arg\min_x (l(\boldsymbol{\phi}(x), \boldsymbol{\phi}_0) + R_{\boldsymbol{\theta}}(x)) \tag{5.2}$$

where $l(\boldsymbol{\phi}(x), \boldsymbol{\phi}_0)$ is a loss function between the feature representation of the input $\boldsymbol{\phi}(x)$ and the target feature representation $\boldsymbol{\phi}_0$. $\boldsymbol{\phi}_0$ can either be the weights of the filter one wishes to visualize, or in this analysis, the final feature vector of the target class. In this case, the loss function is usually defined as the Euclidean distance between the two vectors. Alternatively, although the logic is somewhat circular, the loss function can be defined as the negative of the similarity, typically calculated using a dot product. This analysis will opt for the simpler case defined by Yosinski [**108**].

The optimization can be effectively solved using a gradient descent procedure. The pixels in x are modified in the direction of the gradient of $a_i(x)$. Consequently, the regularization

is usually applied to the gradient step rather than in the objective function itself. Several regularizers are suggested in [**108**] and [**60**], with the overall goal of restricting the visualizations to natural-looking images. Without such a condition, the resulting images will not be semantically interpretable to humans, even if they are reasonable solutions to the optimization. The authors in [**60**] present two bounds on pixel range and variation, which have some corollaries in [**108**]. Some more complex functions that involve pixel shifts and texture regularizers are also presented. There is not a clear consensus on the optimal regularization methods; therefore, this analysis opts for two relatively simple conditions. Pixel changes that fall outside the normal range are clipped, and a 5x5 median filter is applied every four gradient steps. It was experimentally found that these conditions were satisfactory to produce semantically interpretable visualizations.

5.5. New Applications: Feature Evolution and Transfer Learning

At this point, activation maximization as a method for deep visualization has been thoroughly discussed, both in usage as well as in implementation. Yet, there is much untapped potential in this domain. One key assumption that predicates the use of the algorithm is the existence of a fully trained network. This condition is a natural one: it is logical to visualize features after their modifications during training. However, perhaps visualizing the evolution of features during the training process would be even more enlightening. Most observations of neural network training have involved tracking values of loss functions or validation accuracies; now, there is an opportunity to visualize the actual features at play. By visualizing features at several time points during training, the evolution of features can be compared to improvements in performance and shed light on the otherwise obfuscated learning procedure.

This new line of thought also presents the chance to observe another somewhat enigmatic facet of neural networks: transfer learning. As described in [**107**], the generality of low-level features suggests that a network trained on one task may only need to slightly modify those features in order to perform an entirely different task on new data. The authors argue that it is the deep layer features that are task specific and thus require greater changes. In practice, this manifests itself when a standard CNN architecture is initialized with weights from one task and then fine-tuned with a new dataset. It can be seen that the training procedure will converge faster, and in some cases the accuracy may even be higher than if the starting weights were randomly initialized. With this new paradigm of using activation maximizations to visualize features during learning, perhaps a greater understanding of this phenomenon will emerge.

5.6. Results and Discussion

Two experiments were designed to examine visualizations that arise during the training of a CNN. In one instance, the filters weights in the network were randomly initialized in the usual fashion. The other case began with weights trained on the ImageNet ILSVRC 2012 dataset for 1000 class object classification [**76**]. The CNN architecture used in both cases is the VGG-16 network described in [**89**]. The network was implemented in Theano using Keras as a front-end [**8, 12**]. Some additional references were used in the compilation of the code [**98, 11**]. The Adadelta optimizer was used in the training procedure [**110**], and categorical cross entropy was used for classification. The network was trained using an NVIDIA Titan Z, with total training times on the order of several hours. The activation maximization implementation also made use of the Titan Z, where each visualization took 2.5 minutes to complete.

In both instances, the classification task was to differentiate between a small subset of the ImageNet data. Namely, only four classes were used: tree frog, flamingo, pool table, and hamburger. Because there are only four classes in this new task, the last layer of the VGG network was changed from a length of 1000 to a length of four. As a result, the weights from this layer could not be transferred in the pretraining experiment. Each class contains 1300 images, yielding a total dataset of 5200 images. 400 of these images were put aside in a validation set.

5.6.1. Trained from randomly initialized weights

Table 5.1 shows the validation accuracy during training at each epoch. After the training set is passed through the network a single time, the model performs classification on the validation set with 60% accuracy. The model is fully trained after 23 epochs and achieves 93% accuracy at this time. Only epochs in which the validation accuracy increases are shown. The corresponding activation maximizations at each epoch are shown in Figure 5.1. Each image is divided into four sections, each corresponding to one class. The classes, starting from the upper left section in clockwise order are: tree frog, flamingo, hamburger, and pool table.

Epoch	Validation Accuracy
1	60%
2	80%
4	83%
6	88%
8	89%
12	90%
13	91%
18	92%
23	93%

Table 5.1. Validation accuracy during training; no pretraining

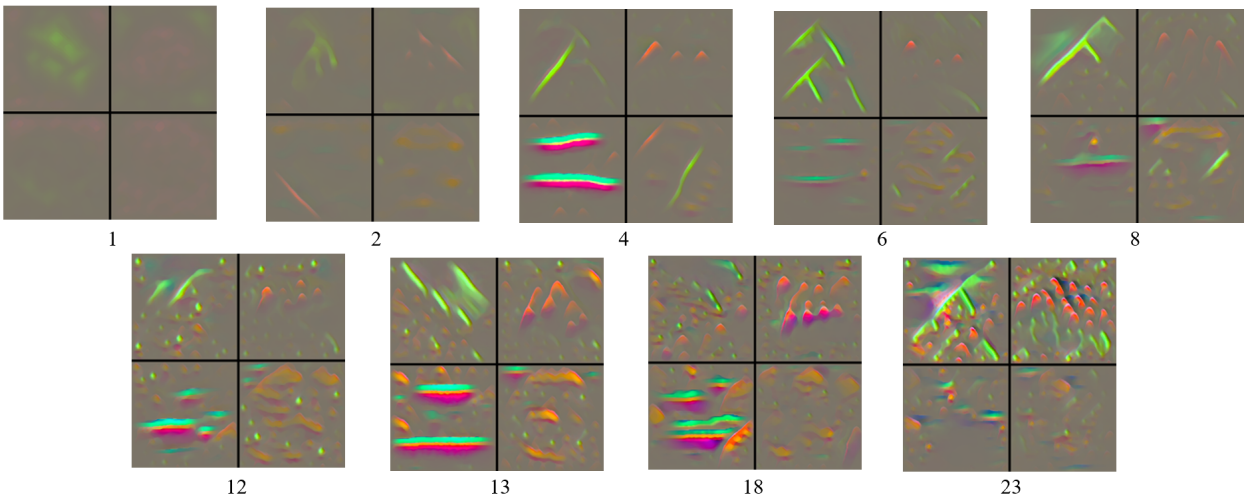


Figure 5.1. Visualizations of network at each training epoch; no pretraining. Classes from upper left (clockwise): tree frog, flamingo, hamburger, pool table.

To begin, it is clear that the visualizations at the early layers of the network are not very informative. At this stage, the convolutional filters have not been fully developed, nor is the validation accuracy high enough to justify their efficacy. That said, within a few epochs one can see some salient features forming. In epoch 4, it appears that the tree frog class is represented by a series of green lines, the flamingo class by some pink shapes, and the hamburger class by similar brown shapes. The pool table class is more strongly defined, with the visualization showing a very prominent horizontal colored line detector. Perhaps this shows an early understanding of the discontinuity between the colored felt of a pool table and the wooden rails. After epoch 12, the validation accuracy exceeds 90% and while the features do appear to increase in complexity, they are not nearly representative of their corresponding classes.

Based on the results in the literature of activation maximization applied to networks trained on the full ImageNet dataset, one would expect the visualizations to more closely resemble the original objects. Figure 5.2 shows the visualizations of such a network; in this

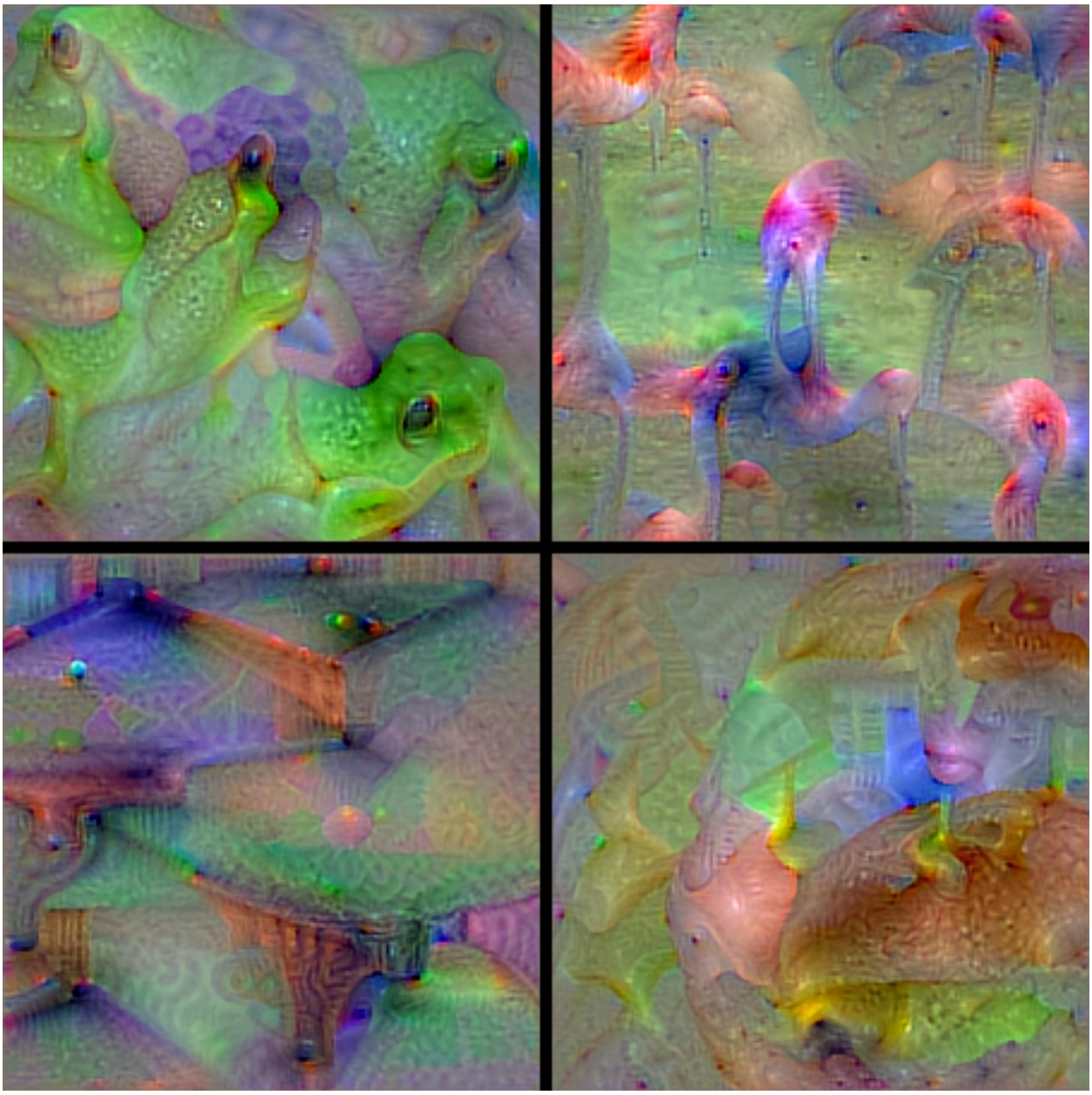


Figure 5.2. Visualization of network trained on full ImageNet. Classes from upper left (clockwise): tree frog, flamingo, hamburger, pool table.

case, activation maximization was applied to the VGG network fully trained on the entire ImageNet set and without any fine-tuning with the small four-class subset. One can clearly see notions of the objects in these visualizations, from frog eyes and flamingo necks to pool

balls and hamburger buns. It appears that the discriminatory power of a feature is heavily dependent on the difficulty of the task: a simpler classification task will yield simpler features even when the constituent data is the same.

5.6.2. Pretrained with full Imagenet

This observation is further supported by the results in Figure 5.3, showing the visualizations of a network pretrained on the full ImageNet dataset. The corresponding validation accuracies at each epoch are shown in Table 5.2. In this instance, the network converges to high validation accuracies sooner than previously. This is to be expected, given the transferred knowledge already in the network. The eventual maximum accuracy is higher, reaching 96% by epoch 8. Again, given the study of transfer learning in [**107**], this result is unsurprising. The features are also more complex; but, yet again, the features are not as complex as in Figure 5.2. It may be argued that the tree frog features resemble eyes by epoch 8, the flamingo shapes are more pronounced, and the hamburger buns are more discernable. The pool table features are much more apparent; in fact, the visualization in epoch 14 does seem to show red pool balls lined up on blue or green felt. Many deductions can be made from these results. For one, it may be argued that the additional accuracy from pretraining is most likely due to the added feature complexity. Furthermore, these results still support the theory that simplifying the classification task will result in less complex and well-defined features.

One final modification was made in this experiment: a visualization was shown for epoch 20, where the validation accuracy slightly decreases. Given that the training loss at this time was still decreasing, this suggests that the network may have begun to overfit the data. It seems that even though the complexity of the features is still increasing, the images

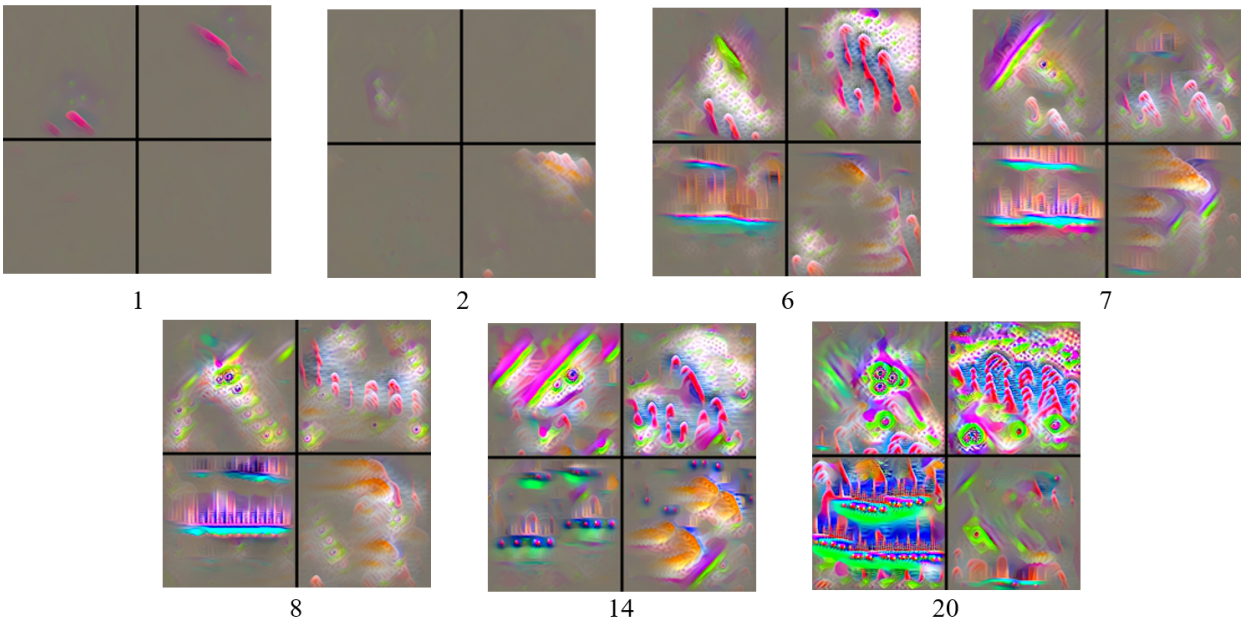


Figure 5.3. Visualizations of network at each training epoch; pretrained on full ImageNet. Classes from upper left (clockwise): tree frog, flamingo, hamburger, pool table.

Epoch	Validation Accuracy
1	59%
2	89%
6	91%
7	95%
8	96%
14	96%
20	93%

Table 5.2. Validation accuracy during training; pretrained on full ImageNet

themselves are less clear. For example, it appears the tree frog eyes that exist in epochs 8 and 14 begin to manifest themselves in the visualizations of other classes by epoch 20. One particularly notable case is in the bottom section of the flamingo visualization. In addition, the pink neck shapes that define the flamingo class appear in both the tree frog and pool table visualizations. Perhaps this confusion of features is an illustration of the mechanism

behind which overfitting can degrade the discriminative power of a network. More testing would be required to fully investigate this phenomenon.

5.7. Summary

Deep visualization of feature evolution, especially in the case of transfer learning, is a nascent approach to understanding CNNs. In this work, activation maximization was used to experimentally justify several arguments. First, feature complexity increases with validation accuracy, but can continue to increase even after accuracy saturates. Also, the discriminative classification power of a network is a function of the number of classes; i.e. a CNN automatically generates features of just enough complexity to perform the task at hand, even when the network is pretrained on a more challenging task. Additionally, training on a more challenging task (e.g. larger number of classes) will yield features that are more informative and archetypal of the representative class members. Finally, unchecked feature complexity leads to feature confusion, a potential precursor to overfitting. In the following chapter, we will see how activation maximization can also be used as a tool for comparing features between different types of networks; namely, capsule neural networks and CNNs.

CHAPTER 6

Examining the Benefits of Capsule Neural Networks

6.1. Chapter Abstract

Capsule networks are a recently developed class of neural networks that potentially address some of the deficiencies with traditional convolutional neural networks. By replacing the standard scalar activations with vectors, and by connecting the artificial neurons in a new way, capsule networks aim to be the next great development for computer vision applications. However, in order to determine whether these networks truly operate differently than traditional networks, one must look at the differences in the capsule features. To this end, we perform several analyses with the purpose of elucidating capsule features and determining whether they perform as described in the initial publication. First, we perform a deep visualization analysis to visually compare capsule features and convolutional neural network features. Then, we look at the ability for capsule features to encode information across the vector components and address what changes in the capsule architecture provides the most benefit. Finally, we look at how well the capsule features are able to encode instantiation parameters of class objects via visual transformations.

6.2. Introduction

Convolutional neural networks (CNNs) have long been the tools of choice when tackling computer vision problems. The spatial localization of CNN features is greatly beneficial when the networks are applied to images and videos; however, these networks also have their

shortcomings. The kernels in a convolutional layer must learn to identify the presence of all relevant features in the input. Thus, transformations such as rotations and occlusion can be detrimental when the training dataset is not properly augmented. Even still, the burden of learning visual features in addition to all possible modifications of these features can be immense for a traditional CNN.

Recently, a novel class of neural networks was proposed in [**77**] that employs the concept of a "capsule". The authors describe a capsule as a group of neurons that represent the existence of a feature in addition to parameters regarding the instantiation of said feature. Contrary to the scalar activations of kernels in a traditional CNN, these capsule vectors aim to be richer representations of information in the network. In this manner, a capsule should be able to encode not only the existence of a particular visual feature, but also the transformations it can undergo in the given application.

That said, while initial results show great potential for capsule networks, there is still much uncertainty regarding how these capsules function. In fact, the "black box" analogy can be applied to all classes of neural networks, not just those with capsules. The interpretability of neural networks has always been a problem, and it is difficult to examine the benefit of capsules without a comparison to traditional CNN features.

In an attempt to elucidate these capsules, this investigation will begin by employing a deep visualization technique to generate images that visually represent the information contained in a capsule. This image can then be compared to an image created in a similar fashion from a traditional CNN, and the discrepancies between them can provide visual justification for the hypothesized benefits of capsule networks. Furthermore, the visual impact of modifying values in a capsule are examined to more accurately ascertain their capacity. Finally, the investigation will examine other facets of the original capsule network

architecture proposed in [**77**], namely the benefits of dynamic routing and a reconstruction network. The next section highlights related work in the field, followed by an outline of the capsule network and visualization methodologies. Finally, the results are shown and the resulting trends are discussed.

6.3. Related Work

The concept of a capsule neural network originated in Hinton's 2017 paper [**77**], wherein the capsule vectors are described and implemented within a convolutional architecture. Furthermore, a dynamic routing algorithm is proposed that selectively links units in a capsule together rather than traditional downsampling methods such as max pooling. There is a follow up publication from Hinton in 2018 [**32**] that extends capsules to matrix form as well as further developing the routing scheme; however, our work will primarily focus on the architecture discussed in [**77**], and our experiments will be in parallel to those performed in the first publication.

Other modifications to the original architecture have also been proposed, including in [**104**] where the capacity of the network is increased (both via numbers of layers and size of capsules) along with changes to the activation function. The authors in [**74**] demonstrate that capsules without the masking operation used in [**77**] may generalize better. The work in [**105**] extends the capsule scheme to a multi-scale hierarchy. A generative adversarial network (GAN) is proposed in [**40**] that makes use of capsules in the discriminator network. The network in [**15**] takes hyperspectral images as input as opposed to standard RGB images. The authors in [**67**] create a Siamese capsule network by combining pairwise inputs with the capsule architecture.

The applications for capsule networks have also been widespread. In [**55**], a capsule network uses images taken by a UAV for classification of rice fields. A detection problem is performed to find street signs in [**47**], while the authors in [**44**] use capsules to analyze traffic patterns in a city. The work in [**36**] outlines a capsule network for seagrass segmentation in satellite images. Video data is used as input to an action detection network using capsules in [**17**]. Capsules have also been used on text data for classification [**75**] and sentiment analysis [**103**]. The authors in [**4**] design a reinforcement learning approach with capsules to play complex games.

Despite their relative nascency, many have started using capsules in the medical domain, including for segmentation [**49**] and cancer detection [**65**] in lung CT scans. These networks have also been used on MRI data for brain tumor classification [**2**] and histology images for breast cancer identification [**35**]. The authors in [**42**] discuss challenges of using public medical datasets in the context of capsule networks. Finally, [**6**] proposes a spectral capsule network to solve the "learning to diagnose" problem.

Clearly, these capsule networks exhibit great potential; yet, the justification for how these networks perform so well is less clear. Granted, Hinton enumerates several potential benefits of capsules in [**77**], namely that the increased dimensionality of the capsules allows feature transformation encoding and that dynamic routing is a more intelligent way of aggregating information. That said, the experimental results, while impressive, are not necessarily proof of that the capsules are exhibiting these traits. One set of experiments in [**77**] seems to indicate that certain object features can be controlled via capsule manipulation, but this is not explored to greater depth. The authors in [**84**] make a more concerted attempt at explainability by varying output capsules in more than one dimension, but yet again this methodology is somewhat limited in scope.

Our investigation compounds on these capsule manipulation experiments by adding deep visualization techniques. These techniques are aimed at creating images from a trained network that represent the information contained in the weights. From this, one can gain greater insight into how a model functions and what features are in use. The primary technique we employ is activation maximization, which is described in [**59, 18**] and generalized in [**60**] to describe other techniques, including inversion [**16**]. While we will not extend much beyond the activation maximization structure, other investigations have looked at CNN features with attention maps [**108, 111**] and saliency maps [**88**]. The authors in [**69**] use activation maximization and a clustering approach to visualize different "facets" of image classes, the visualizations are performed while training in Chapter 5, and the Google deep dream generator [**3**] uses a procedure similar to activation maximization to create art.

Fundamentally, the application of activation maximization to a capsule network for the purpose of understanding the benefits over a traditional CNN is a nascent investigation. Moreover, given that the justification for capsule networks at a feature level has not been thoroughly explored, the necessity for understanding capsules before adopting them in the field is paramount. In the next section, we describe activation maximization and the other methods we employ for the purpose of analyzing capsule features.

6.4. Methodology

To begin, our investigation applies the deep visualization technique of activation maximization to two trained neural networks: a capsule neural network, and a CNN with comparable computational power and information capacity. By comparing the resulting images, we are able to distinguish the different feature representations in these two networks and

glean insight into the potential benefits of capsules. The second experiment further scrutinizes the capsule features in order to more directly ascertain whether capsule vectors truly model transformation parameters. This is done by applying a principal component analysis (PCA) on a set of manually transformed images. The resulting PCA spaces indicate structure in the capsule vectors related to the respective transformations. Then, to even further demonstrate the transformation encoding ability of capsules, a modified activation maximization procedure is used to generate images that correspond to said transformations. By modifying capsule vectors along the principal components to varying degrees and then using the modified activation maximization procedure, we can see that the capsules can generate images with varying degrees of visual transformations. Finally, these investigations are performed when the reconstruction network that is typically present in the capsule network architectures is removed. Some results are also shown in the case when dynamic routing is removed. This section will outline the capsule network architecture, activation maximization algorithm, and how these are used in conjunction with PCA to perform energy compaction and transformation encoding on capsule vectors. Specific results for these methods will follow after some experimental details.

6.4.1. Capsule Network Architecture

This investigation employs an architecture identical to the one outlined in [**77**], as shown in Fig. 6.1. The network takes as input a 28 x 28 grayscale image and proceeds with a standard convolutional layer with ReLU activation, followed by a strided convolution layer. At this point, the feature maps are split into groups before being reshaped into the primary capsule layer. The nonlinearity used for the last step is the "squash" function developed in [**77**] and

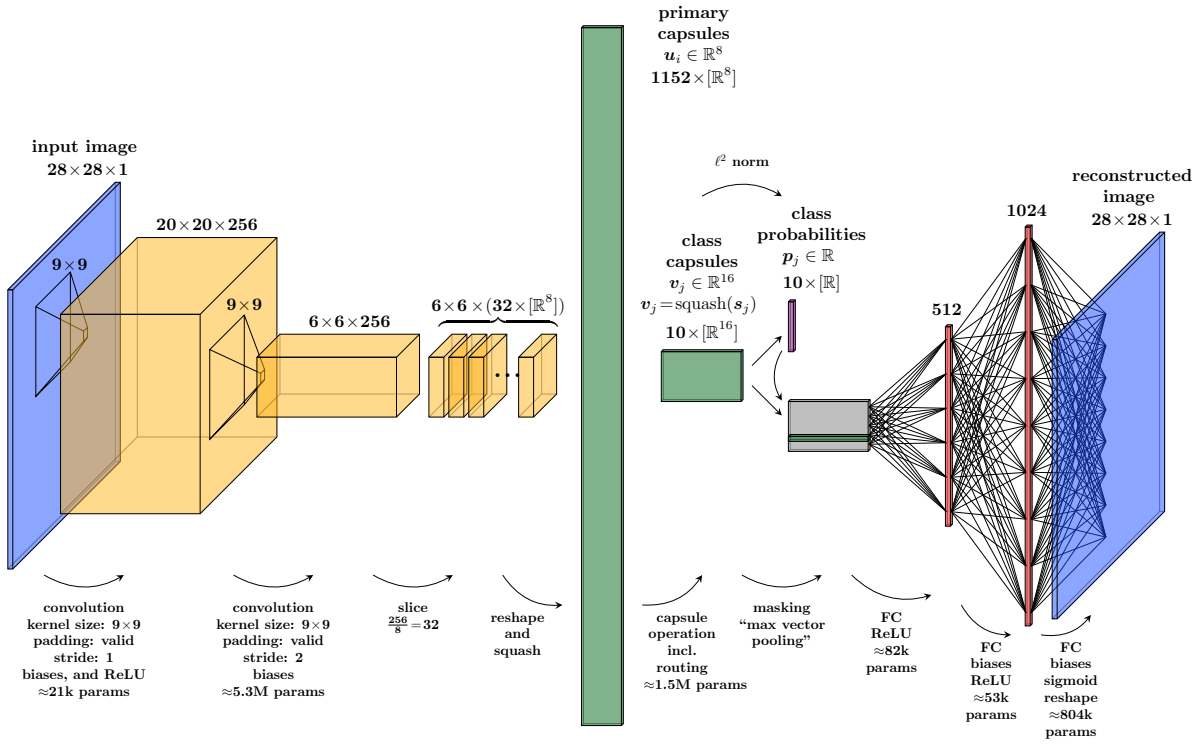


Figure 6.1. Capsule network architecture including reconstruction network used as regularizer

defined by:

$$\mathbf{v}_j = \frac{||\mathbf{s}_j||^2}{1 + ||\mathbf{s}_j||^2} \frac{\mathbf{s}_j}{||\mathbf{s}_j||} \tag{6.1}$$

where $\mathbf{v}_j \, \epsilon \, \mathbf{R}^8$ is the vector output of the capsule and $\mathbf{s}_j$ is its input. This activation function aims to maintain the direction of a capsule vector while normalizing its length such that short vectors are mapped to vectors with near zero length while long vectors are mapped to vectors with length close to one. The class capsule layer follows the primary capsule layer, and it is at this point where the dynamic routing algorithm is implemented. This "routing by agreement" serves as a more advanced method of neuron connection as compared to traditional methods like max-pooling which can lose all but the most prominent connections. Again from [**77**],

the capsule input is given by:

$$\mathbf{s}_j = \sum_i c_{ij} \hat{\mathbf{u}}_{j|i} \tag{6.2}$$

with

$$\hat{\mathbf{u}}_{j|i} = \mathbf{W}_{ij} \mathbf{u}_i \tag{6.3}$$

where $\hat{\mathbf{u}}_{j|i}$ are the prediction vectors found by multiplying the capsule vectors in the previous layer $\mathbf{u}_i$ by the weight matrices of the layer $\mathbf{W}_{ij}$. The coupling coefficients c_{ij} used in the dynamic routing process are given by the "routing softmax":

$$c_{ij} = \frac{\exp b_{ij}}{\sum_k \exp b_{ik}} \tag{6.4}$$

where b_{ij} are the logits of the coupling coefficients that are iteratively refined by the routing algorithm as proposed in [**77**]. The initial logits are set to zero in all our experiments. In doing so, the coefficients converge towards agreement of the output of one capsule $\mathbf{v}_j$ with the output of a capsule in the previous layer $\hat{\mathbf{u}}_{j|i}$.

After the class capsules are found, the l^2 norm of $\mathbf{v}_j$ is used to find the class probabilities, which in turn are used to make the final classification. While this is the entirety of the network at testing time, it is trained with a reconstruction network that takes the output of the largest capsule vector (corresponding to the classification label) and applies three fully connected layers. The output of these layers is the same size as the reconstructed image, and the mean squared error of this image and the input of the total network is used as an added term in the loss function. This reconstruction network acts as a method of regularization to ensure that the capsules maintain sufficient information to represent the input. With the network defined, we now describe the activation maximization method performed on a trained capsule network.

Algorithm 1: Routing Algorithm

begin

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: $\mathbf{c}_i \leftarrow \text{softmax}(\mathbf{b}_i)$

for all capsule j in layer $(l+1)$: $\mathbf{s}_j \leftarrow \sum_i c_{ij}\hat{\mathbf{u}}_{j|i}$

for all capsule j in layer $(l+1)$: $\mathbf{v}_j \leftarrow \text{squash}(\mathbf{s}_j)$

for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i}\mathbf{v}_j$

end

end

Figure 6.2. Routing Algorithm

6.4.2. Activation Maximization

After the network is trained, activation maximization can be used as a means of visualizing the features learned by the network. In general, activation maximization is an optimization approach to produce images that can represent either intermediate network features or object classes. It can be formulated as:

$$x^* = \arg\max_x (a_i(x) - R_{\boldsymbol{\theta}}(x)) \tag{6.5}$$

where x^* is the final visualization, x is a candidate input image to the network, $a_i(x)$ is the activation for some particular unit i, and $R_{\boldsymbol{\theta}}(x)$ is some parameterized regularization function. Depending on the choice of the unit i, the visualization represents different kinds

of information. In a CNN, if i is chosen to be the index of a filter in a convolutional layer, the visualization will depict an image that corresponds to the maximum output of the filter. Depending on the choice of filter, these visualizations could manifest as object components or texture patterns. If, on the other hand, i is chosen to be an element in the final layer class probability vector, the visualization will depict the aggregation of network features that most strongly represents the class. In other words, these images should be the most optimal exemplars of the class. We will be employing this functionality on both a CNN as well as on a single element in the final layer of a capsule network.

We will also use a slightly modified version of this procedure. To begin, we restructure the problem from a maximization to a minimization:

$$x^* = \arg\min_x (l(\boldsymbol{\phi}(x), \boldsymbol{\phi}_0) + R_{\boldsymbol{\theta}}(x)) \tag{6.6}$$

where $l(\boldsymbol{\phi}(x), \boldsymbol{\phi}_0)$ is a loss function between the feature representation of the input $\boldsymbol{\phi}(x)$ and the target feature representation $\boldsymbol{\phi}_0$. If $\boldsymbol{\phi}_0$ is chosen to be a one-hot indicator vector for a given class, then the result of this optimization is the same again an exemplar image from the class. However, one can also choose a different $\boldsymbol{\phi}_0$, such as the capsule vector found by passing a particular image through the network. This would create an image that very closely resembles the input image. As such, it is more well defined as an "activation matching" procedure rather than the traditional title of activation maximization. We use this technique later when analyzing the transformation encodings.

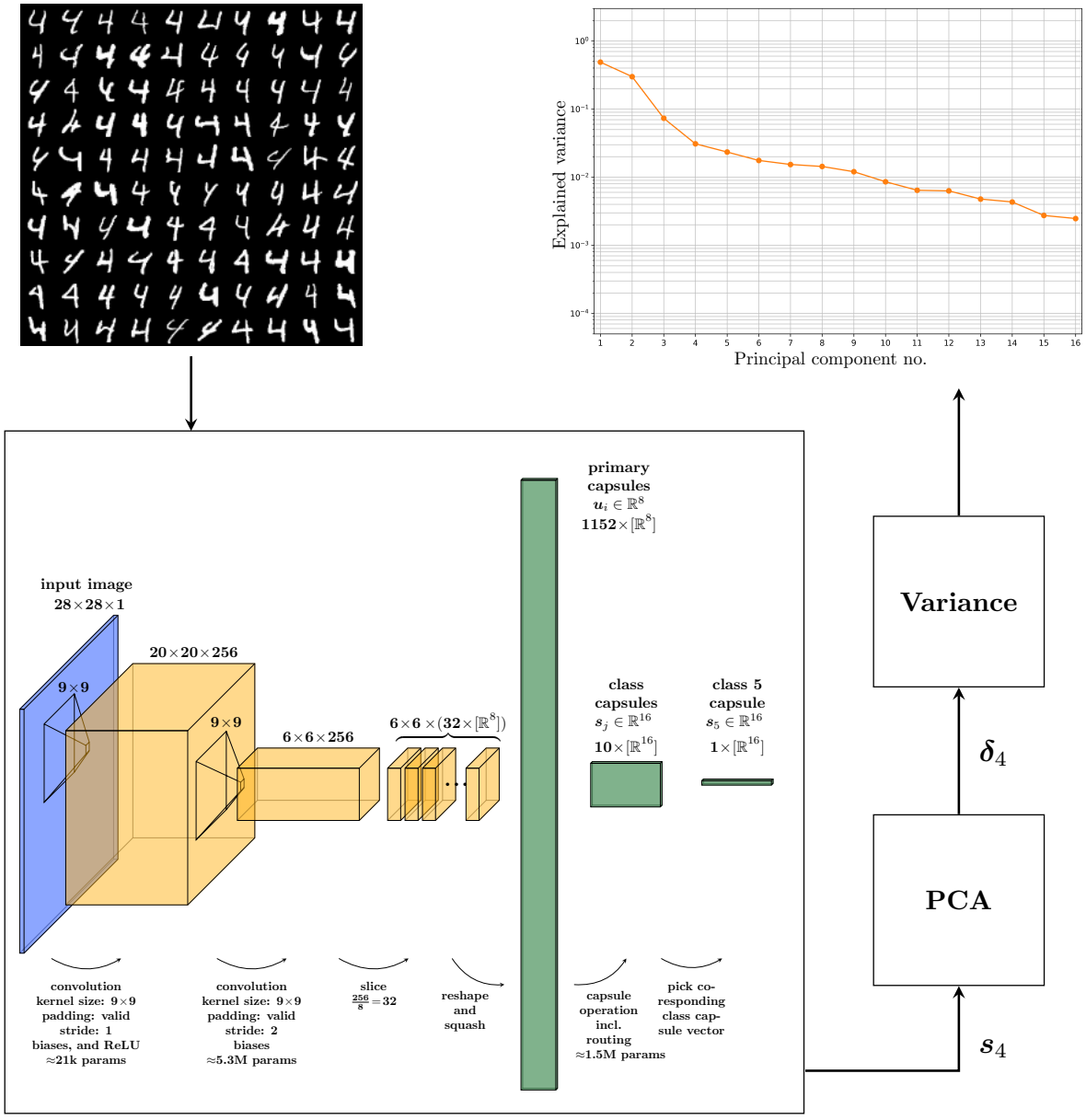


Figure 6.3. Energy compaction for set of images

6.4.3. Energy Compaction

While the activation maximization images provide a method to visually examine network features, the energy compaction analysis outlined here presents a more quantitative comparison. After the capsule network is trained, a set of images is passed through it in order to

obtain the corresponding capsule vectors. Principal component analysis (PCA) is performed on these vectors, following which the variance along each of these particular dimensions is found. One can see a diagram of this pipeline in Fig. 6.3.

This variance is a measurement related to the distribution of energy or information along a specific principal axis. If the information is heavily concentrated in one particular direction in the native capsule space, the variance along the first principal component will be disproportionately large. This in turn indicates that the number of dimensions required to represent the information is small, with perhaps even just one dimension being sufficient. In this investigation, this phenomenon correlates with capsule vectors that do not encode information across all dimensions in the capsule space; rather, these vectors only encode information in a small number of dimensions. Because the capsule vectors proposed in [**77**] are supposed to store instantiation parameters, this may indicate that the capsules are not functioning optimally. Thus, the benefit over traditional CNN features may also be limited.

The converse is also true: when the information in the capsule vector is well distributed, the variance along the principal axes will be more balanced. Granted, any PCA will yield components that have decreasing variance as the component number increases; however, the slope of this trajectory is more gradual in this case. Consequently, many if not most of the components are required to represent the information in the capsule vectors. This potentially correlates with capsule vectors that are functioning optimally and suggests a benefit over CNN features in line with the findings in [**77**].

6.4.4. Transformation Encoding

To take the energy compaction analysis one step further, we perform a transformation encoding analysis that uses the developed PCA-based framework to examine how capsule vectors

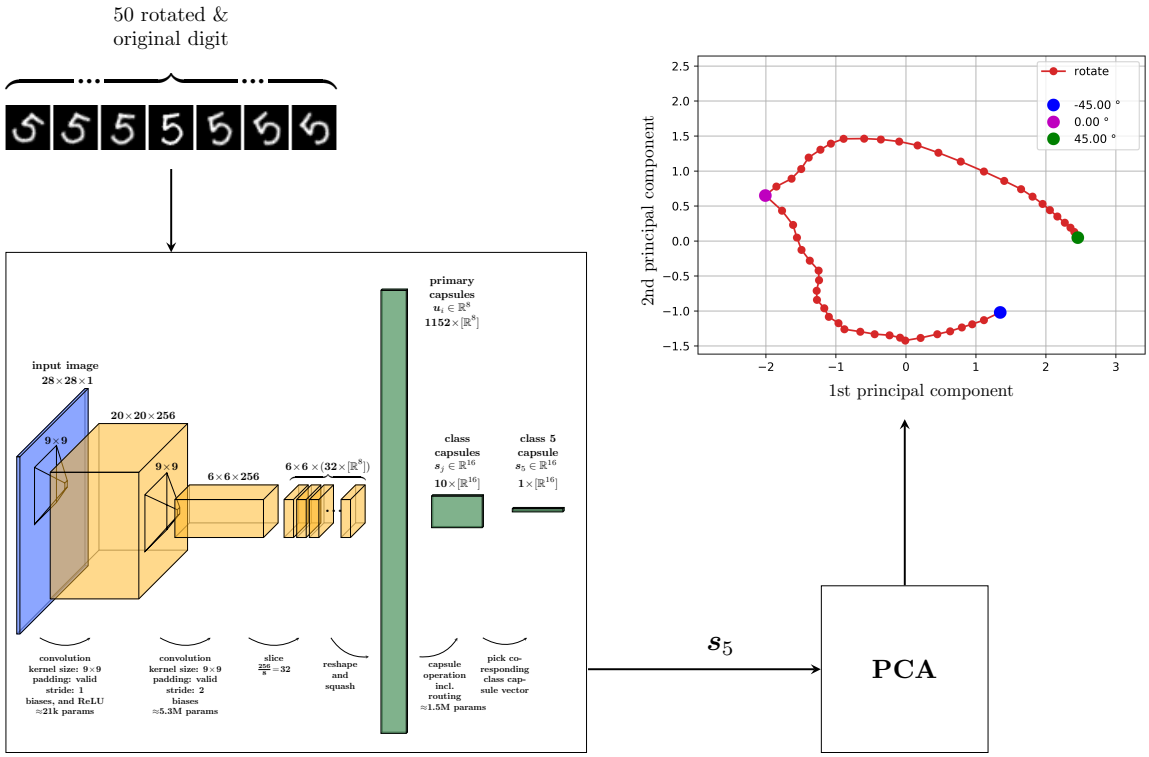


Figure 6.4. Transformation encoding forward analysis for rotation

encode image transformations. The authors in [**77**] claim that capsule vectors should be able to encode image transformations such as rotation and scale changes. In order to examine this claim, we perform both a forward analysis and a pre-image analysis.

6.4.4.1. Forward Analysis. The forward analysis begins with a manually generated set of transformed images. One can see an example of this in Fig. 6.4, where an image from the MNIST dataset (shown in the center of the row of images) is manually transformed with varying degrees of rotation. The resulting images are used as input to the energy compaction procedure which yields principal components as before. Instead of plotting the variance as a function of principal component index, the plot in Fig. 6.4 shows each image as a single point on a 2D grid spanned by the first and second component values. For example, the

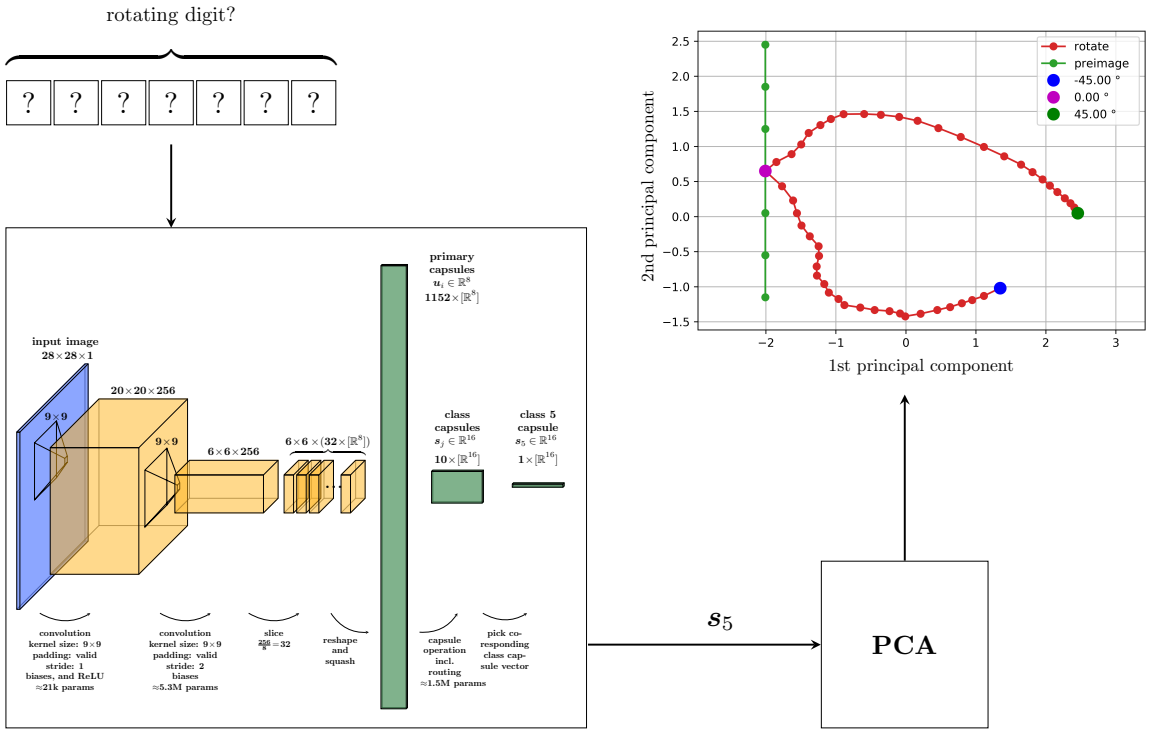


Figure 6.5. Transformation encoding pre-image analysis on generated PCA space

original image (purple point) has a first principal component value of approximately equal to -2 and a second principal component value of approximately equal to 0.7. The points are linked to show how the images span the principal component space with increasing transformation intensity. The green and blue points in the plot show the images with the largest transformations, which in this case are the images with 45nd -45otation, respectively. One can glean some insight from observing the shape of these curves; for example, a smooth transformation curve that is oriented with principal component axes is indicative of relative organization in the capsule domain. However, the more explanatory results are shown when the forward analysis is followed by pre-image analysis.

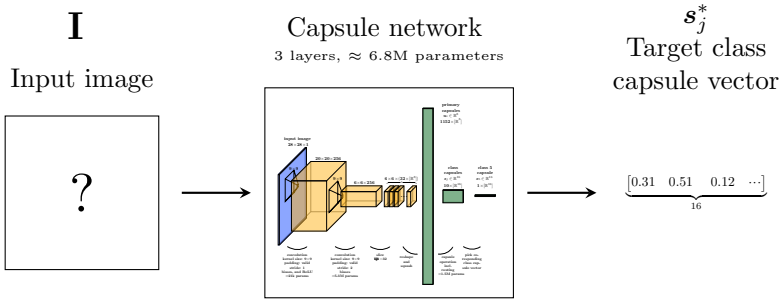


Figure 6.6. Activation maximization/matching for pre-image analysis

6.4.4.2. Pre-image Analysis. The term "pre-image" has been used in several ways in the deep visualization literature. Here, we use the term to define an image whose capsule vector most closely matches a particular target. Continuing the example described in Fig. 6.4, we see in Fig. 6.5 that the pre-image analysis aims to find images that match some target values when passed through the network and whose capsule vectors are then transformed into PCA space. While the objective of the forward analysis was to generate transformation curves in the PCA space, the objective of the pre-image analysis is to ascertain the ability to control visual transformation via capsule vector modification. The ability to do so further justifies the claims made in [**77**]. In Fig. 6.5, the green points represent modified capsule vectors in the PCA space. In this particular case, the original digit image without rotation (shown in purple) had the second principal component value modified with varying degrees to form a set of pre-images. A modified activation maximization procedure, perhaps more accurately described as activation "matching", was used to find the corresponding pre-images and is shown in Fig. 6.6. This procedure is formulated in almost the same way as before; however, instead of minimizing the loss between the feature representation of the input and an indicator function, the target was chosen to be the modified class capsules vectors. After performing this optimization, one will find images similar to those used in the forward

analysis. These images can then be used to ascertain the capsule vectors' robustness to image transformations.

6.5. Results and Discussion

The capsule network was trained on the MNIST dataset of handwritten digits in the same manner as [**77**]. Therefore, the ten capsules in the final capsule layer each correspond to a particular digit. When capsule vectors need to be isolated for a particular class in either the activation maximization or PCA-based procedures, the respective row of the capsule matrix is taken for further processing. The network itself was implemented in Tensorflow and trained on a single NVIDIA Tesla P100 GPU. The Adam optimizer [**45**] was used with the originally proposed decay rates and the resulting training times were approximately 15 hours when routing was used and 12 hours when routing was omitted. A baseline CNN architecture with similar computational cost was also trained on the MNIST data using the specifications outlined in [**77**].

In the activation maximization and pre-image algorithms, two forms of regularization were used; first, a median filter of kernel size 3x3 was applied every 100th gradient step and second, pixel values outsize the normalized 0 to 1 range were clipped at each step. These two regularization methods ensured that the resulting images were interpretable and stayed within the distribution of the original dataset. The following sections describe and discuss the results from each of the previously formulated methods using this experimental setup.

6.5.1. Comparison of Classification Methods

After training, the classification error rates of the capsule network and CNN were 0.28% and 0.49%, respectively. In Table 6.1, one can also see the classification performance of

Model	Configuration		Error Rate (%)
	Reconstruction	Routing	
Baseline CNN	-	-	0.49±0.027
CapsNet	no	no	0.34±0.020
CapsNet	weak	no	0.33±0.030
CapsNet	strong	no	0.33±0.024
CapsNet	no	yes	0.39±0.019
CapsNet	weak	yes	0.31±0.031
CapsNet	strong	yes	**0.28**±0.017

Table 6.1. Classification Error Rates for Network Configurations (5 Trials)

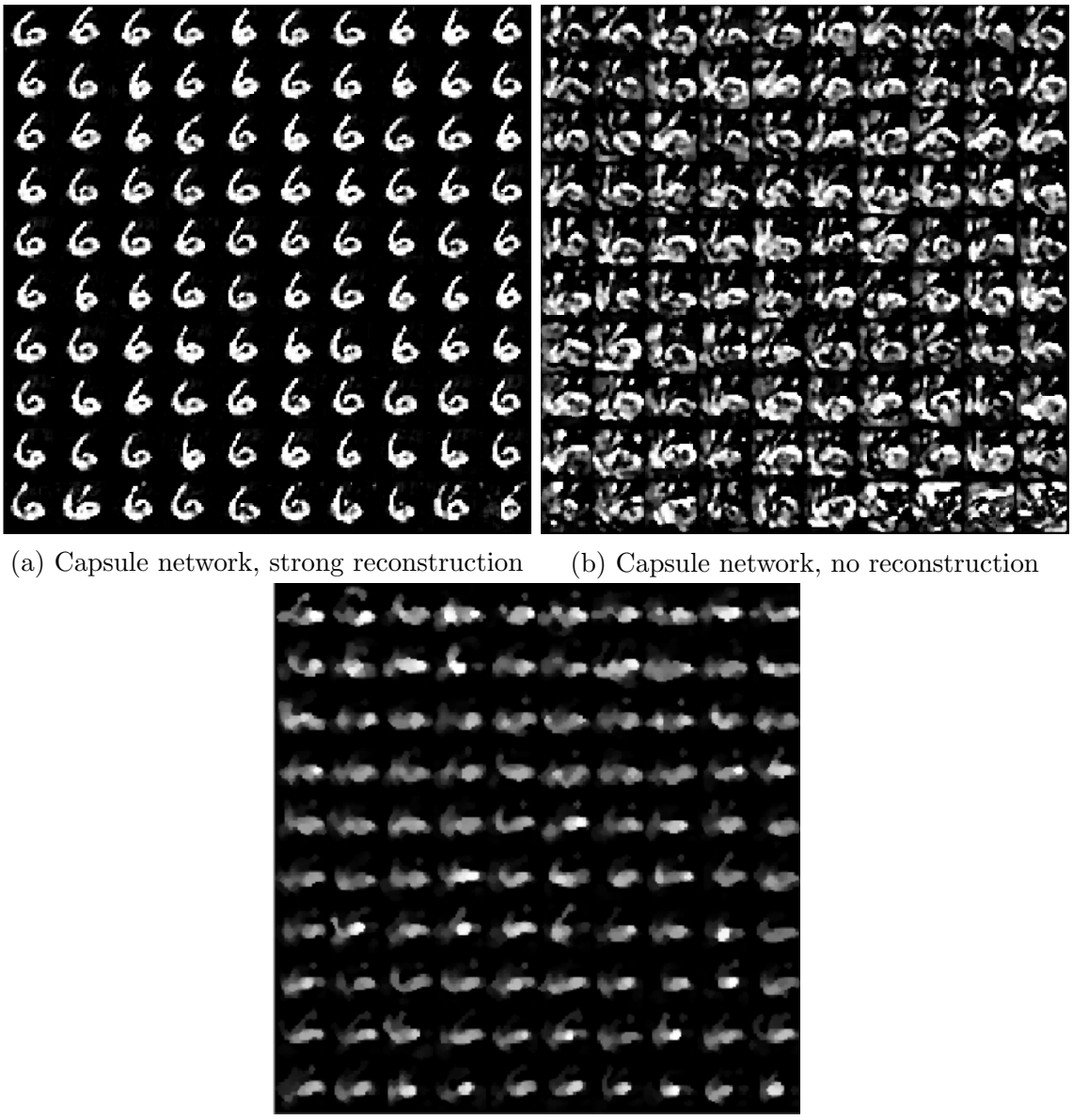
the capsule network configurations with varying amounts of the reconstruction regularizer and dynamic routing. The dynamic routing algorithm can simply be turned on or off. In the the latter case, the capsules are still structured as previously described; however, the routing coefficients are not iteratively modified as in the algorithm. The reconstruction configuration is defined as "no", "weak", or "strong". When the network does not use the reconstruction component, the corresponding term in the loss function is set to zero. In the "strong" case, the term weight is 20 times larger than in the "weak" case. From the table, one can see that the capsule network outperforms the baseline CNN in all cases, and furthermore the addition of strong reconstruction and routing does improve the performance. It is important to note the relative importance of each of these components. In the case when no routing is used, the reconstruction network had minimal impact on performance. When routing is used, increasing the weight of the reconstruction loss reduced the error rate. This indicates that, while the proposed dynamic routing contribution in [**77**] does have benefits in a capsule network architecture, the relative benefit of the reconstruction network should not be understated. Without such regularization, the dynamic routing alone does not necessarily provide a benefit as it may even hinder classification performance. With all this said, one may point out that the classification margins are very slim between all these cases given that

the networks all exceed 99% classification accuracy on the MNIST testing set. As a result, other means of comparison are necessary to obtain an accurate picture of the salutary effects of capsules. With this in mind, the activation maximization results are a first step in looking more deeply at capsules.

6.5.2. Activation Maximization

As discussed, the activation maximization analysis aims to create images that represent information learned by a network. Fig. 6.7 shows 100 such images created from the activation maximization algorithm when applied to a capsule network (both with and without the reconstruction network) as well as 100 from the baseline CNN. The images are stacked and ordered in a 10x10 grid by decreasing activation value; thus, the top left image has the highest activation value of the 100 trials while the bottom right image has the lowest value. Recall that the activation maximization images represent the aggregation of features that the network has learned to represent the particular class. From the images in Fig. 6.7a, we can see that the visualizations are very indicative of the class in question. All of the images show the defining characteristics of a "6" digit; that is, both the circular loop at the bottom as well as the upward curving tail. This shows that the capsule network has learned these facets of the class and use all of them when performing classification.

This is in contrast to the activation maximization images from the CNN, as shown in Fig. 6.7c. In this case, the features that the network makes use of are much less clear. Generally, it can be seen that the CNN has some general oblong shapes in the lower half of the images that are likely related to the circular loop of a "6". That said, the clarity of these loops are far worse than those of the capsule network. This supports the notion that the CNN is only searching for an oblong loop in the bottom of the image to classify a

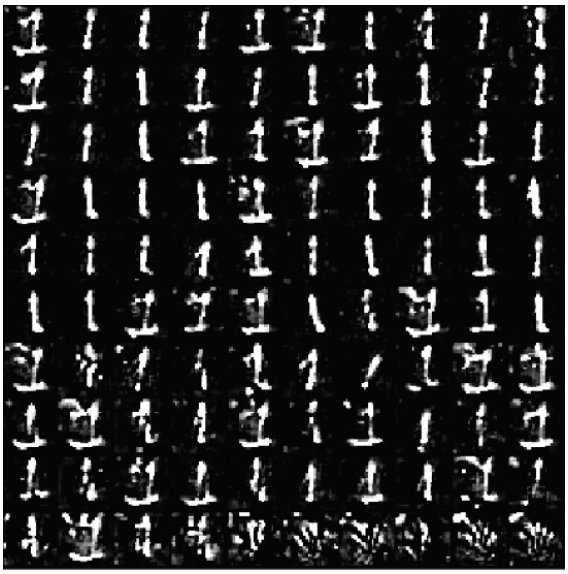


(a) Capsule network, strong reconstruction (b) Capsule network, no reconstruction

(c) CNN

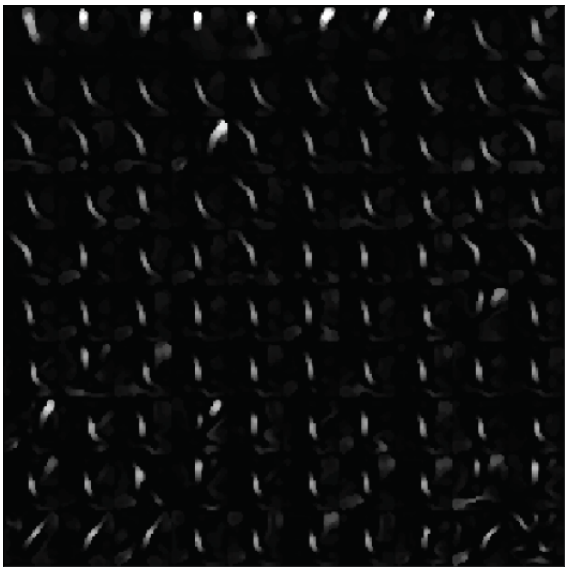
Figure 6.7. Activation maximizations for digit 6

"6". Given a CNN's proclivity to find the lowest complexity feature required to discriminate between classes, this phenomenon is fairly logical. That said, it is consequently important to look at this distinction in the context of the differences between the capsule network and the CNN. Both networks perform the task extremely well, as each of them obtains over 99%



(a) Capsule network, strong reconstruction

(b) Capsule network, no reconstruction



(c) CNN

Figure 6.8. Activation maximizations for digit 1

classification accuracy. Thus, the differences in the visualizations can be attributed to feature complexity rather than just classification power. Therefore, there are indications that the capsule network features capture more information from a class than its CNN counterparts.

Fundamentally, the capsule features demonstrate an understanding of the class exemplars to a much greater extent than a CNN feature, which is aligned with the ideas in [**77**].

The images made from the capsule network without reconstruction further paint the picture of how the capsule features appear. In 6.7b, one can see that the visualizations are much less interpretable than in the strong reconstruction case. While the features are potentially visible, they are masked by a large amount of noise. Given that the reconstruction places emphasis on training a network to have the ability to invert features back into the original image, it is natural that the capsule network without reconstruction would have much less interpretable features. Furthermore, the distinction is in line with the classification error rates in Table 6.1, where the capsule network without reconstruction performs worse.

When looking at other classes, one sees similar results. In Fig. 6.8, the same procedure is performed on the "1" digit class. The capsule network visualizations have a few more artifacts than in the "6" case, as one can see in the last row, but the general trend still holds. The capsule features are much more descriptive and representative of the members of the dataset class whereas the CNN features are very minimalist. In this particular case, we see that the CNN features are primarily vertically oriented lines. This is, of course, logical for the class in question, but it also neglects the potential distinctions between members within the class. Some "1" images are just vertical lines, others may include a vectored top, and others still have the horizontal underline. Here, the visualizations indicate that the capsules are able to codify intra-class variability to a greater extent than CNN features, which again follows the rationale of [**77**].

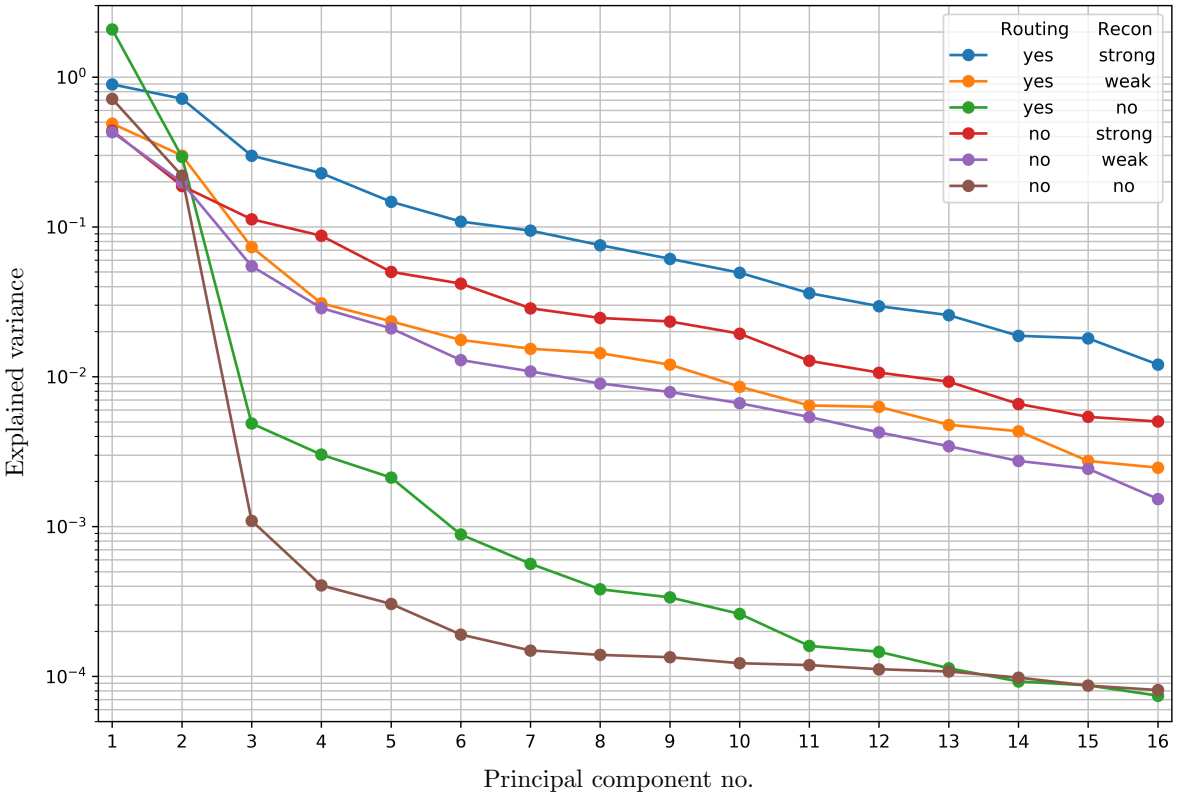
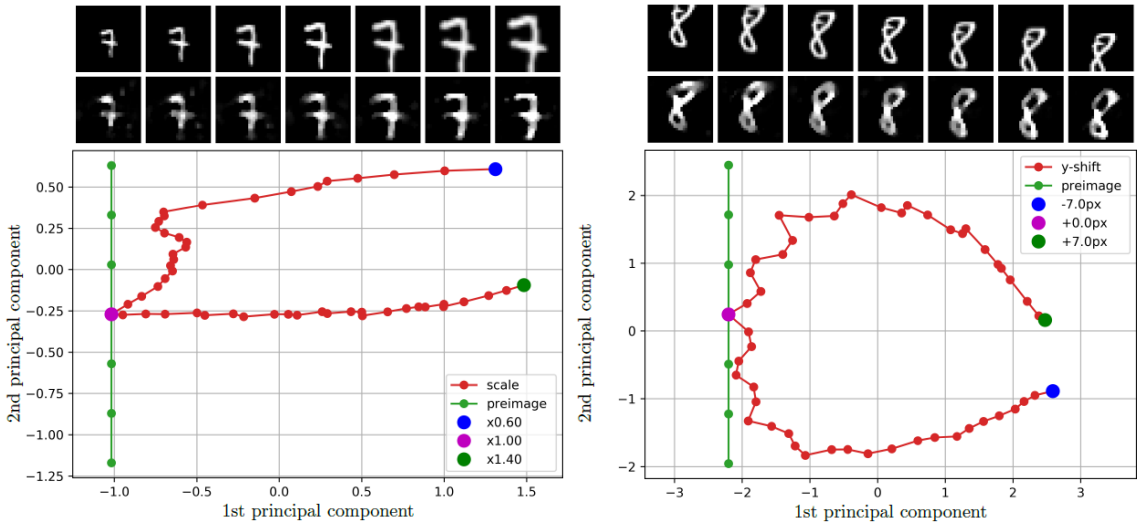


Figure 6.9. Energy compaction analysis on different network configurations

6.5.3. Energy Compaction

The activation maximization analysis, while enlightening in its own right, is somewhat limited by its qualitative nature. Thus, the described energy compaction analysis provides a quantitative foil to the visualization results. All of the images in the testing set are passed through the trained network and the resulting capsule vectors are extracted. Then, PCA is performed on the set of capsule vectors. From here, the variances for each of the components are calculated and are shown in Fig. 6.9. Each line corresponds to a capsule network that has varying amounts of reconstruction and routing, as was the case in the classification error comparison.

To reiterate, the variance of a particular PCA components corresponds to the relative amount of information that is contained in that vector dimension. Naturally, the variance decreases for each subsequent principal component; however, the rate of decline is indicative of how much the information is spread amongst the capsule dimensions. An effective capsule vector has a gradual variance decline over its components as the information is distributed effectively, while an ineffective vector has the vast majority of information contained in the first few principal components. Looking at Fig. 6.9, we see that the network with the most gradual decline includes routing and strong reconstruction. On the other hand, the network with the sharpest decline is the one without reconstruction or routing. This indicates that the features in the no reconstruction and routing network are not as information rich, which is consistent with the classification error and activation maximization results as well. Furthermore, we can see that regardless of the presence of routing, increasing the weight of reconstruction improves information distribution. In fact, comparing the routing vs. no routing networks for each reconstruction scenario, we see that the curves are relatively close. The difference is non-negligible, as the routing curves are consistently higher than the non-routing curves; yet, the gap between these two curves is small in all reconstruction cases. Moreover, changing the reconstruction strength yields a bigger change in the curves, especially when the reconstruction is removed altogether. This builds on the results seen in the activation maximization analysis and supports the notion that the presence of the reconstruction loss is pivotal to the overall efficacy of the capsule networks. Additionally, this gives credence to the idea that the reconstruction loss is in fact more important to capsule function than the presence of dynamic routing. This point was perhaps only vaguely alluded to in the classification error comparison, but this energy compaction analysis certainly elucidates it. This concept is not as well explored in [**77**], but is nonetheless extremely important



(a) Scale modification

(b) Vertical pixel shift

(c) Rotation

(d) Dilation and erosion

Figure 6.10. Transformation encoding analysis (forward and pre-image) results

to the understanding and usage of capsules. While dynamic routing does improve the information distribution in capsule features, the reconstruction network is potentially much more important for the desired behavior of the capsules.

6.5.4. Transformation Encoding

To conclude the investigation into capsule features, a transformation encoding analysis was performed in order to examine the ability for the capsule vectors to encode attributes of the application classes. The authors in [**77**] perform a small scale analysis wherein they show that varying a single value in the capsule vector results in specific visual transformations in the reconstructed image. In this manner, the authors explored the impact of the reconstruction network rather than the capsule network itself. For example, modifying one of the capsule values would change the scale of the object in the reconstructed image while another would translate the object. In doing so, the authors claim that this demonstrates that the individual capsule values are able to encode instantiation parameters of class objects, which in turn supports the conclusion that capsules are more robust to these modifications than traditional CNN features. However, this investigation is only a very small (and not thoroughly described) set of experiments. To this end, we performed a more in-depth investigation that consists of both a forward and pre-image analysis. As previously described, the forward analysis takes a set of manually transformed images and performs PCA in order to create a "map" of visual transformations via capsule modifications in the PCA space. Then, a pre-image analysis is done by modifying capsule values in the PCA space and examining visual changes in the image. This is similar to the original experiments in [**77**]; however, in this case the capsule changes are well documented and one can see these changes in the context of distance travelled in the PCA space. Furthermore, because we are showing the changes in the input image space, our results more effectively show the impact of the capsule network rather than the reconstruction network.

In Fig. 6.10, one can see four different variations of the transformation encoding experiment. Each case shows the results of both the forward and pre-image analysis. To begin, a single image from the dataset is manually modified by a particular transformation. These images are shown in the top row; for instance, in Fig. 6.10a, one can see that the original "7" image (center of top row) is scaled both up and down by up to 40%. Only a subset of the total number of created images are shown. These images are taken to the PCA space and plotted by the first two principal components along the red curve. This curve spans the manifold on which the image can be scaled up or down. Then, the second principal component is modified to yield points shown on the green line. The second component was chosen experimentally, as modifying the first principal component did not yield the desired transformation. Rather, the objects lost definition to the point where they did not resemble the original class members. This may point towards the fact that the first principal component controls class identity whereas the remaining control the various instantiation parameters. This follows from the construction of the capsules due to the fact that the capsule lengths are used to determine class identity. If the capsule are primarily used for classification, then the length may be the most important facet of the vectors, and therefore the largest variation that the PCA pulled out could have been by lengths. Regardless, these points are then transformed back into the native capsule vector space and are used to create images via the modified activation matching procedure. These image are shown in the second row of each subfigure.

To begin, we can see from Fig. 6.10a that the capsules are generally able to reproduce scale changes in the image with small amounts of distortion. Similar patterns can be seen when other transformations, such as y-shift and thickness, are modified. Rotation was found to be a more difficult transformation to emulate, potential because the variability of rotation

in the original training set was likely very small. In Fig. 6.10c, one can see from the second row of images that the pre-image analysis was generally only able to slightly rotate small parts of the object. For example, the bottom tail of the "2" is only rotated in the counter-clockwise direction (rightmost images) whereas the top of the "2" remains somewhat stationary. The reverse is true in the clockwise case, where the top part of the object is able to rotate more easily than the bottom. Again, this is likely because the network was not shown images with large rotations during training, so it is unlikely that large rotations would need to be encoded in the capsules.

The distortion in the pre-images generally occurs when the PCA modifications result in capsule vectors that diverge from the original manifold (red curve). This is also quite logical: when the vectors diverge from the original "scale" manifold, it is very likely that other visual changes should occur that may be tangential to simple scaling. In this case, the object lose some of their definition; yet, the quintessential object features remain. This more readily justifies the thesis proposed in [**77**], as one can see that modifying several capsule values in a manner close to the observed capsule changes in the PCA space via the forward analysis gives images that follow the visual trend. This is a more comprehensive view of the transformation encoding power of capsules than in [**77**], as the authors there claim that single capsule values can control each facet. This may not necessarily be the case, depending on the model that results from the training procedure. However, even when this is not the case, this analysis shows that ordered modification of multiple components can result in the same phenomenon. Thus, the capsule vectors do indeed encode instantiation parameters and this can be an asset over CNNs in classification tasks.

6.6. Conclusion

As shown, capsule network features do fundamentally operate differently than CNN features. In the activation maximization analysis, one could see that the capsule features were better able to describe all facets of the class objects than the CNN features. That said, when the reconstruction network was removed from the capsule pipeline, the features degraded and did not have as much discriminative power. In the energy compaction experiment, we showed that capsules with routing and reconstruction were adept at spreading information across all the elements of the capsule vectors. As the reconstruction weighting was reduced, so too did the information become condensed within one of two principal components, which is more in line with how a scalar CNN feature may behave. Finally, the transformation encoding analysis showed that the capsules are indeed able to capture instantiation parameters of class objects, which is a major benefit over CNN features. The sum of these experiments show that capsule features do have the potential to surpass CNN features, but it is important to note that the reconstruction part of the capsule networks is essential for the desired behavior, whereas the dynamic routing algorithm may not be as beneficial.

To further the work started in this investigation, applying these techniques to a more complicated dataset may produce more discernible differences in classification rates. This may obfuscate the ability to compare features, as the better performing network would naturally have more discriminative features, but there may be benefits to having an experiment where the performances of all networks do not exceed 99% classification accuracy. Additionally, given how important the reconstruction network was to capsule performance, it may be valuable to compare these results with a CNN that similarly includes a reconstruction network for regularization. Finally, looking at more advanced capsule architectures, such as

those with deeper capsule connections or with a different routing scheme, would be valuable. In this manner, one could truly ascertain whether these network are in fact the next stage of evolution in solving computer vision tasks with neural networks. Another variant of CNNs will be explored in the next chapter, wherein activation maximization is used with the aim of improving control of the generated output of a GAN.

CHAPTER 7

Improving GAN Controllability with Activation Maximization in the Latent Space

7.1. Chapter Abstract

Generative adversarial networks (GANs) map latent and visual distributions with the aim of generating useful images from scratch. Their effectiveness has improved greatly over the last several years, but the ability to control the visual output of a GAN after the network has been trained is not well established. In this work, we present an algorithm that allows visual modification of generated outputs by modifying vectors in the latent space. This algorithm is shown to be effective in controlling the extent of visual transformations in MNIST images, and is favorably compared to an explainable method that is developed using principal component analysis.

7.2. Introduction

Generative adversarial networks (GANs) have become prevalent in applications involving image generation. GANs produce images from a given distribution, and when trained properly, the results are indistinguishable to the human eye. Yet, a major drawback exists in the relative inability to visually control the created image. For example, a GAN may be trained to generate human faces. The network converts a latent vector into an image that fits within the bounds of a photo-realistic human face; however, once the image is created, there is no way to make modifications to the image while retaining desirable visual traits.

Changes to the latent vector would result in changes in the output image, but a traditional GAN will not have a distribution of latent vectors that is itself interpretable at a human level. In other words, one would not know the relationship between changes in the latent vector and changes in the visual output space. Any new latent vector may yield an entirely different face image that, while still photo-realistic, would have little to no similarity with the original output.

There have been developments in GAN training and architectures that aim to combat this controllability problem. We will discuss several of their respective nuances in the following section, but the majority of them involve conditioning the latent distribution in some fashion. This necessitates training the network with these modifications in place. These developments do not help in situations where the GAN has already been trained and controllability is desired.

In this work, we discuss a method to modify the outputs of a GAN without the requirement of re-training the generator. Fundamentally, we train a network that sits on top of the generator and converts latent vectors of a particular GAN output to those that include a desired visual modification. While this does involve training the latent space conversion (LSC) network, the generator is not touched, and is in fact not used at all in the training procedure. The LSC network is trained from images generated with activation maximization, a tool originally used in the deep visualization field. Deep visualization techniques primarily exist to provide human-level insight into "black box" neural network architectures. In this work, we do so by comparing the results of the GAN LSC to a similar procedure that employs principal component analysis (PCA). Thus, we provide justification to the improved controllability of the GAN. In the next sections, we will survey current works in the field,

describe our technical methodology for accomplishing our goals, and present our findings and conclusions.

7.3. Related Work

GANs were first described by Goodfellow [**26**] in 2014, and work quickly followed that modified the initial architecture to include conditioning on the input latent vector. These conditional GANs describe including an additional vector at the input of the generator. These inputs ranged from class labels or image caption embeddings [**63**] to a vector of attributes for human facial image generation [**22**]. In doing so, not only can one categorical select the types of images that are generated, but also the quality of generated images improves [**70, 10**]. Nonetheless, this is only the first step one can take toward GAN controllability. A less discrete quantization of attribute control can be seen in [**80**], where a rank ordering of images based on transformation extent is performed. When a rich set of attribute labels is available, as in [**28**], one can control attribute presence in generated images in a similar fashion to the previously mentioned works in conditioning. However, we again note that these methods are not functional when such information is not known ahead of time.

Thus, later works began exploration into GAN latent spaces in an effort to extract meaning from vectors in such spaces. The authors in [**56**] perform an optimization to retrieve the corresponding latent vector for a given generated image. This optimization is similar to the activation maximization approach that we will describe later. Some also performed clustering in the latent space [**66**] or latent space separation [**85**] with the goal of enhancing particular attributes of generated images. Perhaps most related to our work is that of Voynov [**100**], in which an optimization is performed to find a matrix that can transform

given latent vectors into those with enhanced attributes. Yet, despite the similarity, the results here are not as promising that those that we will show later on.

As previously stated, our method takes inspiration from deep visualization techniques, namely activation maximization. The technique, laid out in [**18**] and well described by Mahendran in [**59, 60**], is used to generate images that represent an aggregation of learned features in a neural network. Activation maximization has been used to great effect in developing human-level understanding in convolutional neural networks (CNNs) [**108, 111, 69**] and even in artistic pursuits like Google Deep Dream [**3**]. In our previous work [**73**], we use the technique to visualize the training process of a CNN, rather than just the features of the final network. However, our other investigation into capsule networks using activation maximization is the most relevant to this work. There, we examine the benefits of capsule networks in comparison to traditional CNNs by visualizing the respective features with activation maximization. Then, the technique is used to find vectors in the capsule space that correspond to images with particular visual transformations. Thus, a rudimentary mapping of the "capsule space" can be made to examine the function of individual or groups of capsules. This work will perform an analogous latent space map generation, but leverage it more specifically with the aim of improving GAN controllability.

7.4. Methodology

We will now describe the techniques employed in the investigation, beginning with the modified activation maximization approach used to find GAN latent vectors that correspond to particular target images. Then, we discuss using the resulting latent vector-image pairs in a PCA scenario to identify potentially controllable latent vector directions. Finally, the LSC network that improves GAN controllability is described.

7.4.1. Activation Maximization

As previously mentioned, activation maximization is typically used to generate images that visualize attributes of a CNN. To do so, the following optimization is solved:

$$x^* = \arg\max_x (a_i(x) - R_{\boldsymbol{\theta}}(x)) \tag{7.1}$$

where x^* is the final visualization, x is a prospective input image, $a_i(x)$ is the activation for a unit i in the network, and $R_{\boldsymbol{\theta}}(x)$ is some optional regularization function. The choice of i determines the nature of the visualization. For example, if i is chosen to be an element in the vector at the classification layer of a CNN or capsule network, as in Chapter 6, activation maximization generates exemplar images of the chosen class. However, the optimization in Chapter 6 was slightly modified to be:

$$x^* = \arg\min_x (l(\boldsymbol{\phi}(x), \boldsymbol{\phi}_0) + R_{\boldsymbol{\theta}}(x)) \tag{7.2}$$

where $l(\boldsymbol{\phi}(x), \boldsymbol{\phi}_0)$ is a loss function between the feature representation of the input $\boldsymbol{\phi}(x)$ and the target feature representation $\boldsymbol{\phi}_0$. In the previous work, the target representation $\boldsymbol{\phi}_0$ was chosen to be a capsule vector found by passing a particular target image through the network. As such, the activation maximization generated an image that was as close as possible to the target image. Yet, this input image to output capsule vector relationship is not quite analagous to our GAN scenario. Here, the generator takes in vectors as input and produces images at the output.

Consequently, we reorganize the activation maximization optimization as follows:

$$z^* = \arg\min_z (l(G(z), x_0) + R_{\boldsymbol{\theta}}(z)) \tag{7.3}$$

where z^* is the final latent vector, z is a prospective latent vector, $G(z)$ is the output image from the generator given input z, and x_0 is some target image. Now we are able to use this modified activation maximization procedure to find a latent vector that, when passed through the generator, yields an image that is as close as possible to a target image. This effectively allows us to find locations in a GAN latent space for a particular image. In this manner, we can generate multiple latent vector-image pairs to form datasets that are well suited for our controllability analyses.

7.4.2. Transformation Dataset Generation

Figure 7.1. Example manual transformations

We form datasets by repeating activation maximization for different target images. More specifically, we make manual transformations of varying degrees to images in the original training set and perform activation maximization on these augmented sets. These transformations range from pixelwise shifts to rotations, and will be enumerated later. This procedure is similar in nature to what is done in Chapter 6. For example, one may begin with a single image from the dataset and perform varying degrees of rotation to the object as shown in Fig. 7.1. After this augmented set is formed, activation maximization is performed for each image in the set. This process will result in images generated by the GAN that are as close as possible to the images in the augmented set. Additionally, one will have access to the latent vectors that are responsible for creating such images. These latent vector-image

pairs essentially act as anchor points in mapping the latent space of the GAN. The objective, therefore, is to leverage these known points in the latent space to allow controllability with respect to the particular transformation. For instance, in the case of known latent vector-image pairs for rotation, the goal would be to create a system that could perform rotation of an image in the visual space via known manipulation in the latent space. We will demonstrate this controllability with two methods: the first is a PCA approach that is primarily done to give some intuition into how the latent space is organized. The second is a learning-based approach that is shown to outperform the PCA approach and shows promise for more elaborate usage.

7.4.3. Principal Component Analysis (PCA) Investigation

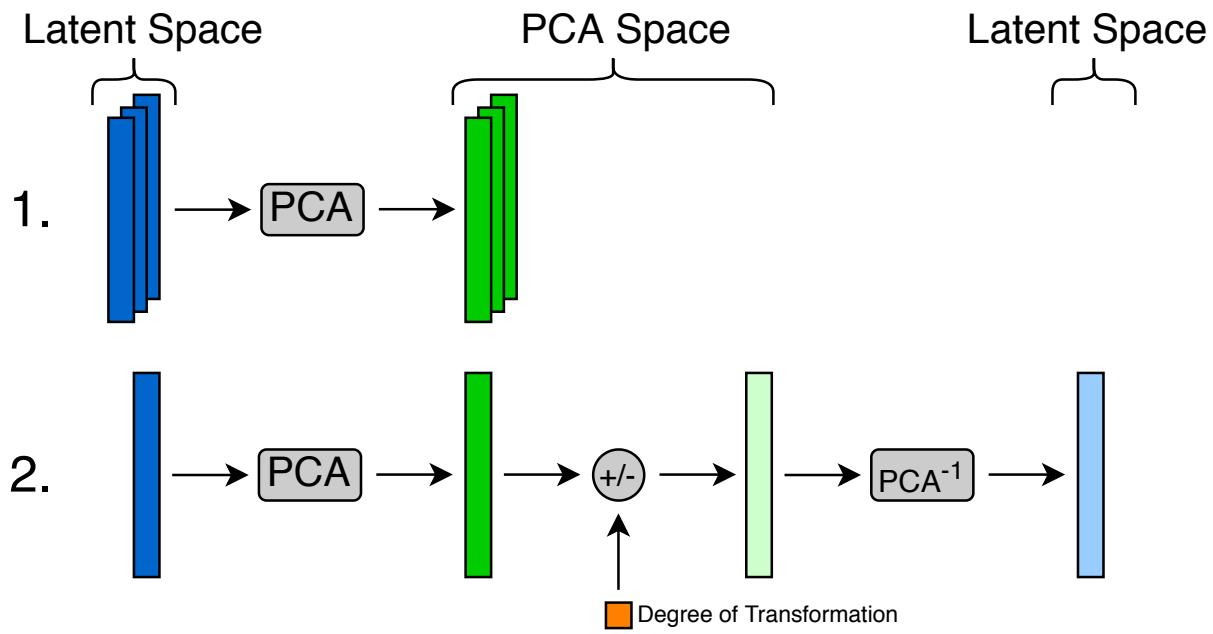
The PCA analysis we perform here takes inspiration again from Chapter 6, but modified to the GAN scenario. The latent vectors found after performing activation maximization on the transformation dataset are aggregated and used to perform PCA. In this manner, we are converting the latent space into a new space that should be more organized with respect to the given transformation. Take the example originally shown in Fig. 7.1: the latent vectors associated with the rotation dataset all correspond to images of approximately the same object with the only differentiating factor being the degree of rotation. One would naturally expect that the differences in the latent vectors to also only correspond to the degree of transformation, as the underlying object still needs to be retained. Thus, performing PCA should result in a space where the first principal components are oriented in the direction of rotation in the latent space. This is identical to the procedure performed in Chapter 6, but with GAN latent vectors instead of capsule vectors. After the PCA transformation is

found, we may test the controllability by modifying the principal components of a particular vector.

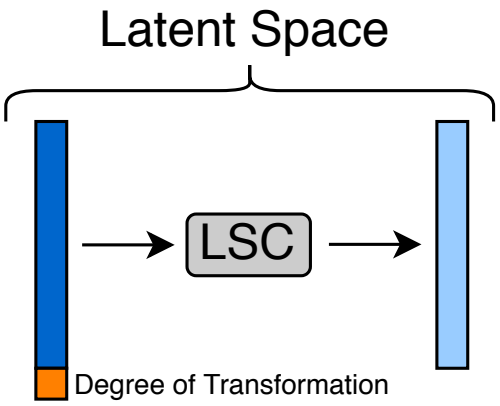
For example, after a PCA transformation is found for the rotation dataset from Fig. 7.1, we may use it to augment one of the images in the dataset. We take the latent vector corresponding to one of the images, such as the original vector for the non-transformed image, and apply the PCA transformation to it. We are left with the vector in the PCA space, which we modify by adding or subtracting to the values in the first principal components. Then, we perform the inverse PCA transform to return the vector to the GAN latent space. At this point, the vector may be sent through the generator to yield the image, which will retain the structure of the original object, but will be rotated by an amount proportional to the value change in the first principal components. This demonstrates the organization of the latent space an the ability to perform transformations without the need of additional conditioning on the generator. Fundamentally, while the PCA procedure demonstrates in an interpretable way how a reorganization of the latent space can be leveraged for increased controllability, it falls short in practicality and visual quality as compared to the learning-based approach that follows.

7.4.4. Latent Space Conversion (LSC) Network

We propose a neural network-based approach in place of the PCA method previously discussed. Instead of the PCA procedure, we train a neural network that takes as input a given latent vector as well as a value corresponding to the desired transformation degree, and outputs the modified latent vector that will yield the modified image when passed through the generator. A comparison of the two approaches is shown in Fig. 7.2. In essence, we remove the need to convert the latent vectors to and from a new space and instead build a



(a) PCA method



(b) LSC method

Figure 7.2. Comparison of PCA and LSC methods

network to perform the task in one step. This latent space conversion (LSC) network clearly simplifies the pipeline, but more importantly we will see that the results are more promising than those of the PCA method. There is a potential loss in explainability, as we introduce a "black box" on top of the generator, unlike the PCA case; however, the ability to produce

more nuanced transformations outweighs this facet and justifies the additional training time as well. In the next section, we outline the details of the network architecture in addition to the other specific design criteria used in our investigation.

7.5. Experimental Design

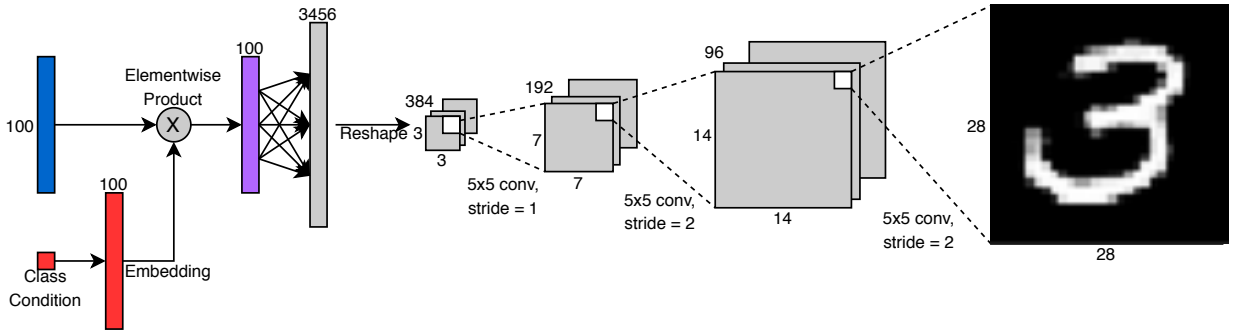


Figure 7.3. Generator Architecture

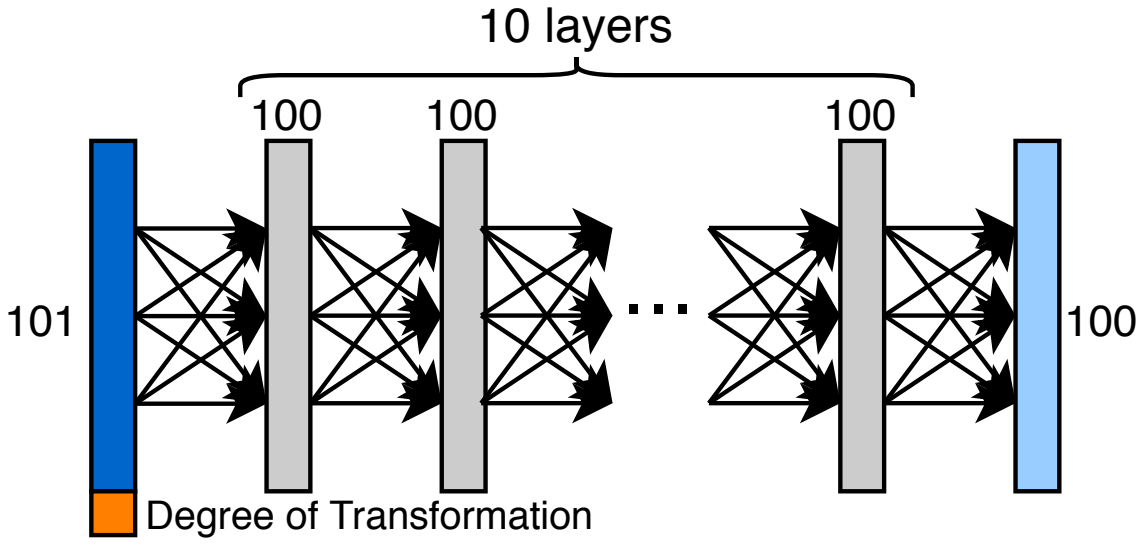


Figure 7.4. LSC Network Architecture

This investigation is generally agnostic to choices in application and network architectures; that is, one could conceivably test this paradigm on a number of different datasets

while using a variety of different GAN and LSC network designs. In this particular case, we opt for some traditional choices as a proof of concept. We begin with a GAN trained on the MNIST digit classification dataset. This training was done previously and can be found as a built-in GAN model in Keras. We take the generator network from this GAN configuration to use in our experiments. The architecture of the generator can be seen in Fig. 7.3, and no additional modifications were made to either the structure nor weights. One can see that the network takes a 100-dimensional latent vector as input and outputs an MNIST image of size 28x28. The generator is constructed as a conditional generator, and as such has an additional input in the form of a scalar class condition that is transformed into a 100-dimensional embedding space and used in an elementwise product with the latent vector before entry into the fully-connected layer in the network. This class condition ensures that the output of the generator will be from the chosen class (Fig. 7.3 shows the case when the class is chosen to be the digit "3"). For our experiments, we fix this class condition and only examine the possible visual changes that can occur within the class. In essence, we only use the "3"-generating function of the network in our pipeline.

Fig. 7.4 shows the architecture of the LSC network that we designed. The stricture is relatively straightforward given the nature of the inputs and output. The network takes in a 100-dimensional latent vector as well as a scalar corresponding to the desired degree of transformation, and outputs a transformed 100-dimensional vector in the latent space. This transformed vector, when passed through the generator, should yield an image similar to the initial image albeit with the chosen visual modification applied to the specified degree. For example, with an LSC network trained to perform rotation, the resulting latent vector should pass through the generator to yield a rotated version of the original image, with a

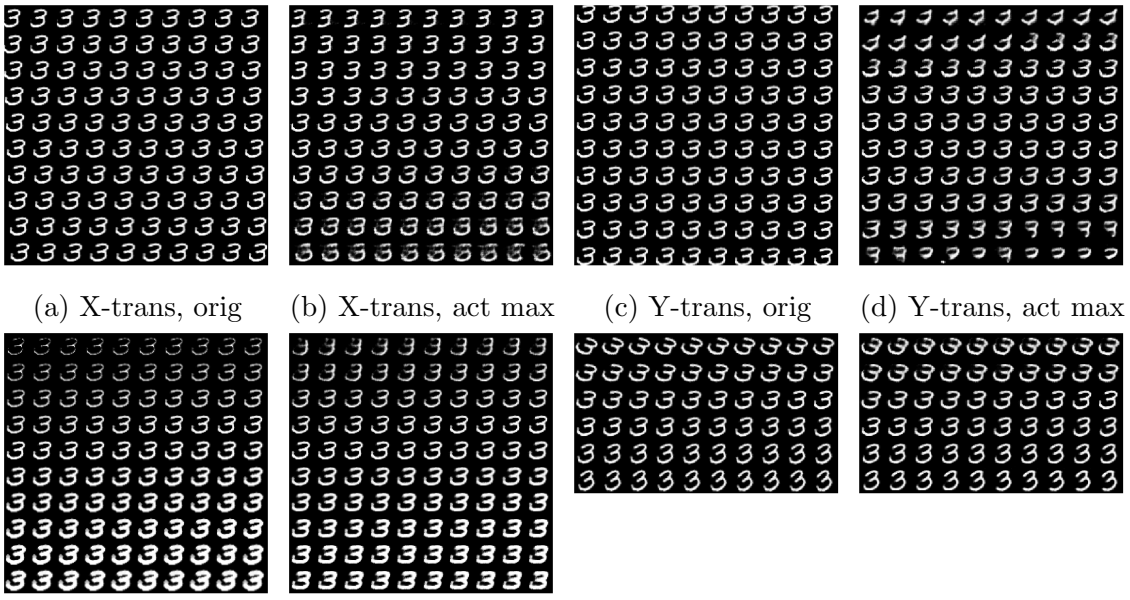
degree of rotation proportional to the transformation degree scalar input. Given the vector-to-vector nature of the problem, we opted for a simple fully connected network with hidden layers of the same size as the output and with 10 layers in total. This was experimentally found to yield effective results. One could, however, replace this LSC architecture with a more complex structure depending on the complexity of the underlying dataset. But again, our particular application did not necessitate the need for any additional complexity.

The networks were constructed and trained in Keras with a Tensorflow backend. The LSC network training was performed on a single Nvidia Titan Z GPU using the Adam optimizer with a learning rate of 10^{-4} [**45**] and a standard mean square error loss function. The activation maximization also used this optimizer with with a learning rate of 10^{-3}. No regularization was necessary for the use of activation maximization on these data. Next, we show the results of these methods and the consequential patterns that emerge.

7.6. Results and Discussion

7.6.1. Activation Maximization Results

We begin by looking at the results of the activation maximization algorithm in its ability to replicate latent vectors that yield visually comparable images to those with manual transformations. In doing so, as previously outlined, a dataset of latent vector-image pairs is created. Without a successful activation maximization procedure, the datasets will be suboptimal and the PCA and LSC approaches will have limited efficacy. Fig. 7.5 shows the comparison between original images and activation maximization images for four different types of transformations. Each image was originally of size 28x28 and was of a single digit. 100 such images are stacked together and shown in each sub-figure. The starting image (i.e. without any transformation) is in the central row, and the images show increasing degrees



(a) X-trans, orig (b) X-trans, act max (c) Y-trans, orig (d) Y-trans, act max

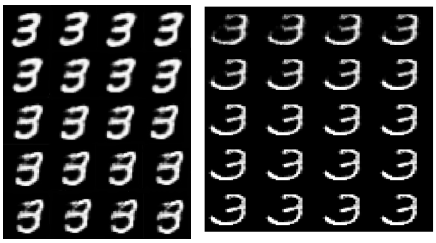
(e) Erosion, orig (f) Erosion, act max (g) Rotation, orig (h) Rotation, act max

Figure 7.5. Comparison of original and activation maximization images

of transformation as one raster scans above and below the central point. From these images, one can see that the activation maximization images are nearly identical to their original counterparts for small degrees of transformation, and are even able to retain the structure at the further transformation extents as well. This is especially remarkable when one considers that the GAN was never trained to produce images with such transformations. In the case of rotation, only a subset of 60 images is shown because the algorithm was unable to effectively produce images with larger degrees of rotation.

7.6.2. PCA Results

Fig. 7.6 shows the results of the PCA method for two of the transformations, x-translation and erosion/dilation. Fundamentally, the PCA results are shown for two reasons. One is to demonstrate a directly interpretable method of GAN controllability, and the second is as a

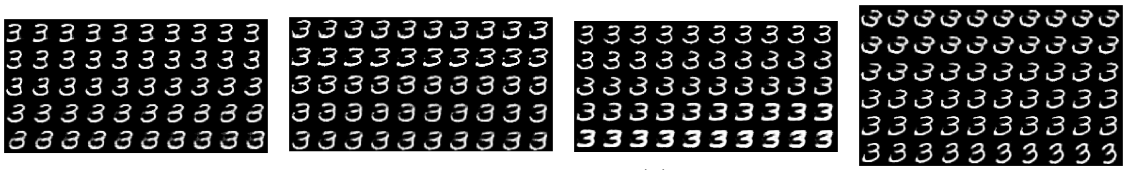


(a) X-trans (b) Erosion

Figure 7.6. PCA Results

point of comparison to the LSC results. Each image is found by adding or subtracting values from the first and send principal components and then transforming the vector back to the latent space and passing it through the GAN. As in the activation maximization results, the top left sub-image in each grid is found by subtracting the largest value from the principal components, while the bottom right arises from the greatest addition to the components. There are only 20 results shown for each transformation due to the lack of sensitivity found in the PCA space.

In general, one can see that the PCA method was not effectively able to capture the underlying transformations as was done in Chapter 6. While the methodology was the same, it is possible that the capsule features in Chapter 6 were more well suited to capturing the variance along the transformation direction in the PCA space. Here, one can see that the x-translation images are not even able to retain the structure of the original "3" image. The erosion/dilation results are slightly better, in that the structure is more or less retained, but the transformative ability is lacking. One may see some slight erosion in the top row of images in the grid, but these minuscule visual changes highlight the inability for the PCA method to adequately capture the necessary transformations. This method is highly interpretable, in that it is clear that moving in the direction of transformation in the PCA space should yield changing images if enough variance is captured by the components. However, it is possible



(a) X-trans (b) Y-trans (c) Erosion (d) Rotation

Figure 7.7. LSC Results

that not enough variance can be captured, and a more sophisticated approach is necessary to effectively control GAN output.

7.6.3. LSC Results

Fig. 7.7 shows the results of the LSC network for all four types of visual transformations. Again, only a sampling of images are shown in each sub-grid, but the LSC network allowed for much more precise control over the output than the PCA method. As a result, we are able to show many more images across the spectrum of visual transformation. Immediately, one can see the marked improvement over the PCA method. In each case, the method is able to retain the structure of the starting image and primarily changes the shape only in accordance with the target transformation. Granted, there can be some changes to the underlying structure in the more extreme transformation cases, say for example at the most dilated and eroded versions of 7.7c, but these changes are not to the extent seen in the PCA case. One can also see the varying degrees at which the visual transformations can be controlled, especially in the more visually apparent transformations of erosion/dilation and rotation. In this manner, it is clear that the LSC methodology has the capacity to control GAN output for the prescribed changes. Furthermore, given the lack of efficacy in the PCA algorithm, this demonstrates the ability for a neural network to learn vector conversions to traverse the latent space of a generator in a more sophisticated way than the

linear directional walks in the latent space that the PCA method employs. This in turn suggests that the organization of the latent space is one in which complex and potentially non-linear trajectories must be plotted in order to span the axis of visual transformation.

7.7. Conclusion

In this work, we present a method for controlling the visual output from a GAN without the necessity of conditioning during training. The algorithm was shown to have efficacy in manipulating images in the visual space along specific axes of transformation without dramatic changes to the underlying structure. Furthermore, the method was shown to excel in comparison to a PCA-based method, suggesting both that the latent space of the generator has complex organization and requires more than linear movement in the latent space to correspond to desired changes in the visual space. With this in mind, one can imagine more complex visual tasks in which this image editing methodology may be applied. However, more work is required in order to generalize this pipeline for more complex transformations as well as allowing the method to function without requiring training for every base image. Some developments to the LSC network architecture may allow for such improvements, such as including a feature term in the loss function or adding more sophisticated connections between the dense layers in the network. Fundamentally, the results are a promising start in a line of work that can truly bolster the usefulness of GANs and generative methods as a whole.

CHAPTER 8

CAMERA: Class Activation Maps for Exemplar Region Attention

8.1. Chapter Abstract

Activation maximization and class activation maps (CAMs) are powerful tools in the field of deep visualization. In this chapter, we present an amalgamation of the two methodologies that amplify the efficacy of their explainability. We present results in the application of Alzheimer's classification using a 3D CNN with MRI input. The results indicate that the network identifies some of the patterns previously discovered in the medical literature, and illustrate how one can combine the fields of biomedical deep learning with feature visualization to further the efficacy of such algorithms.

8.2. Introduction and Related Work

In previous chapters, we have discussed learning on neuroimaging modalities for computer aided diagnosis of Alzheimer's disease, as well as deep feature visualization methods for a variety of different neural networks. However, we have yet to put these two concepts together. One can surmise that the aforementioned feature visualization techniques may have a place in the realm of biomedical machine learning in the hopes of increasing the explainability of traditionally "black box" algorithms. More specifically, activation maximization as we have described it is directly applicable to any convolutional neural network solving a classification task. While we have presented architectures that perform both classification and regression

in chapters 2 and 3, respectively, we will focus here on applying activation maximization to the binary classification network shown in Fig. 2.2.

We will begin with a direct application of activation maximization as discussed in 5. The resulting visualizations can be interpreted on their own; however, we also present a nascent additional step in a pipeline to create even more descriptive visualizations. We will combine the activation maximization algorithm with class activation maps (CAM) [**114**], another visualization technique. Rather than generating interpretable visualizations from scratch, CAM generates a heatmap for a given input into the network. This heatmap can be overlaid upon the input image to show regions in the input that are most contributory towards the CNN decision. In other words, the CAM shows a map of where the network is paying the most attention. One potential drawback of CAMs is that they are input dependent, and could be drastically different depending on which input is chosen. This is where the activation maximization procedure comes into play. By creating an exemplar image of a particular class with activation maximization to use as input into CAM, one can have more confidence in the generalizability of the patterns seen in the attention map. Furthermore, the CAM provides a sense of which parts of the activation maximization images are most responsible for the high prediction confidence. This mutual benefit suggests that the combination of these two methods have the potential to improve the efficacy of both, especially in applications where patterns in the visualizations are not immediately clear.

8.3. Methodology

The procedure for activation maximization has already been discussed at length, and one can refer to any of the explanations in the previous 3 chapters for an overview. Yet, there are a few modifications that are required to adapt to the Alzheimer's task. First, the

CNN is 3-dimensional, so naturally the input will be need to be a volume of the appropriate size. Furthermore, we do not start the optimization with a monochrome or noise image as we have done previously. Experimentally, it was found that the activation maximization procedure was unable to converge towards an interpretable brain shape. The solution to this problem is to begin the optimization with an image that is closer to the eventual endpoint, i.e. beginning with a real brain volume.

While this does solve the problem of convergence to an interpretable image, there are some potential caveats. Depending on the choice of brain volume, one may bias the results towards one of the two diagnoses (healthy or AD). For example, one may choose to start the optimization with a brain from a perfectly healthy individual that has exhibited no evidence of cognitive decline. This particular individual is likely to be classified as healthy with a high probability. As a result, it will not take many voxel modifications to arrive at a brain volume that maximizes the healthy class. On the other hand, starting the optimization towards maximizing the AD class with this same brain will require a great deal more visual modification. This presents an inherent bias whereby the activation maximization of one class may appear more "tampered with" than the other.

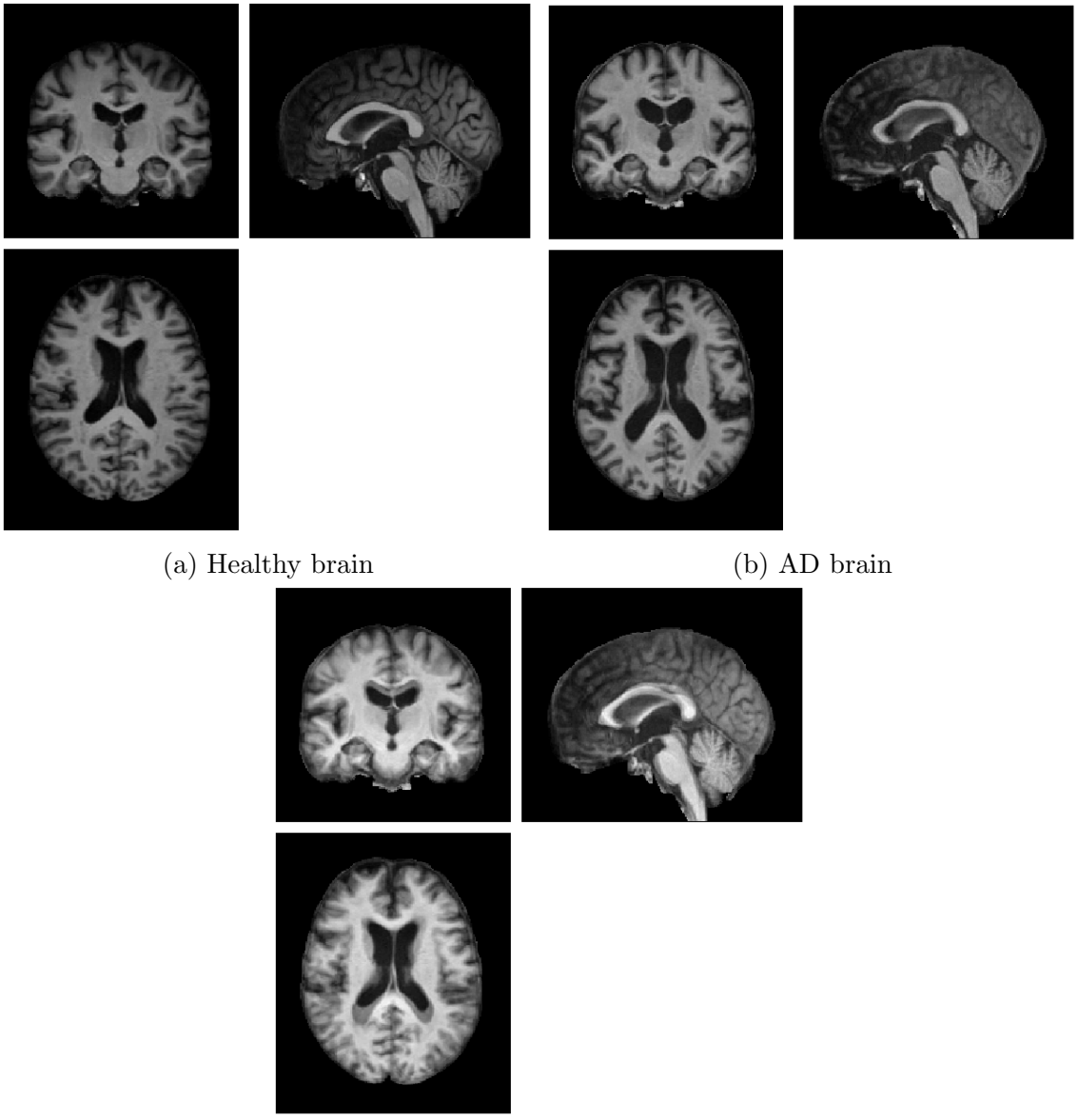
There are several avenues for potentially dealing with this problem. One could opt to start the optimization from a standard brain atlas, such as the MNI152 template used in the pre-processing pipeline, as this "brain" does not necessarily belong to any one individual. However, brain atlases would represent an aggregate of healthy brains, not an aggregate of healthy brains as well as cognitive deficient brains. One may therefore want to use the average brain template formed from the conglomeration of all brains in the dataset, similar to one of the intermediate results of the MRI/PET pre-processing pipeline. But, this again poses a problem. The average brain loses much of the clarity in the folds of the gray and

white matter, and the lack of detail is compounded in the activation maximization to yield images that do not possess the definition to be interpreted.

Consequently, we opt for a measured approach of hand crafting an input brain from the combination of only two brains: one from a patient with a perfect MMSE score and no signs of cognitive decline, and one from an individual at the lowest end of the cognitive spectrum, scoring only 9 out of 30 points on the exam. It also happens that one patient is female and the other is male, so we account for any gender bias in this framework as well. In this manner, we preserve a large amount of detail, but still account for potential differences in the visual distance between the exemplar images. The original and combined brain volumes are shown in Fig. 8.1.

With the starting point of the activation maximization set, one can then proceed to create two visualizations: one for maximizing each of the possible binary classes. Thus, the process yields two images: a healthy exemplar and an AD exemplar volume. The activation maximization procedure is performed with 100 gradient ascent steps with a learning rate of 8000 and a 3x3x3 median filter applied every 10 gradient steps as a means of regularization. At this point, we separately send these two images as input back into the CNN and perform CAM. More specifically, we opt for a Grad-CAM procedure [**83**], which does not require retraining the network with a global average pooling (GAP) layer.

In CAM, as outlined in [**114**], a GAP layer is inserted between the final convolutional and fully connected layers. The GAP layer reduces the tensor of features maps down to a vector of the same length as the number of feature maps in the previous layer. This vector is then fully connected to the output layer, which allows an interesting relationship to develop. Because each unit after the GAP layer results from the corresponding convolutional feature map from the previous layer, one can directly interpret the weights between this layer and the



(a) Healthy brain

(b) AD brain

(c) Combined Brain

Figure 8.1. Original healthy and AD brains, and resulting combined brain volume

output layer as the relative weighting of the particular activations from the feature maps. Thus, if an image is passed through the network and the activations for each kernel are calculated in the deepest convolutional layer, one can multiply the weights from the GAP layer by each activation and sum them together to yield a heatmap that exhibits localization

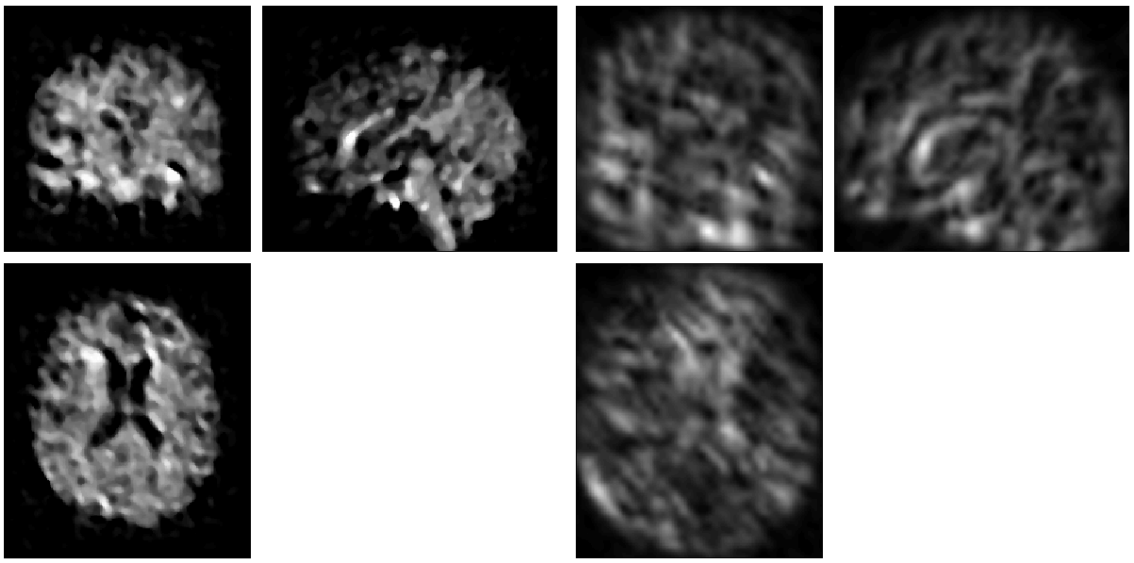
of the most important features (after resizing the activation maps to the original input image size).

The primary issue with CAM is the necessity for a GAP layer directly after the final convolutional layer and before the final output layer. Barring the ability to build this into the architecture, one can use Grad-CAM to yield the same manner of heatmaps with any arbitrary network architecture. In Grad-GAM, the method is more or less the same except for one key difference: the weighted sum is performed on gradients as opposed to the output of a GAP layer. For instance, one may send an input image into a network and calculate the gradients of a particular output class with respect to the activations in the last convolutional layer in the network. These quantities stands in for the activations before the GAP layer. Then, after averaging over all pixels in each feature map, one is left with a vector of the same length as the number of kernels in the convolutional layer. These weights stand in for the weights found after the GAP layer in CAM. Thus, a weighted sum can be performed as before, and the resulting quantity is passed through a ReLU activation function before yielding a CAM heatmap.

With this in mind, we take our two activation maximizations from before and use them as input to the network while we perform Grad-CAM. This results in two heatmaps that map onto activation maximizations to show the relative attention the network paid to specific regions of the model. We therefore describe this procedure as Class Activation Maps for Exemplar Region Attention (CAMERA).

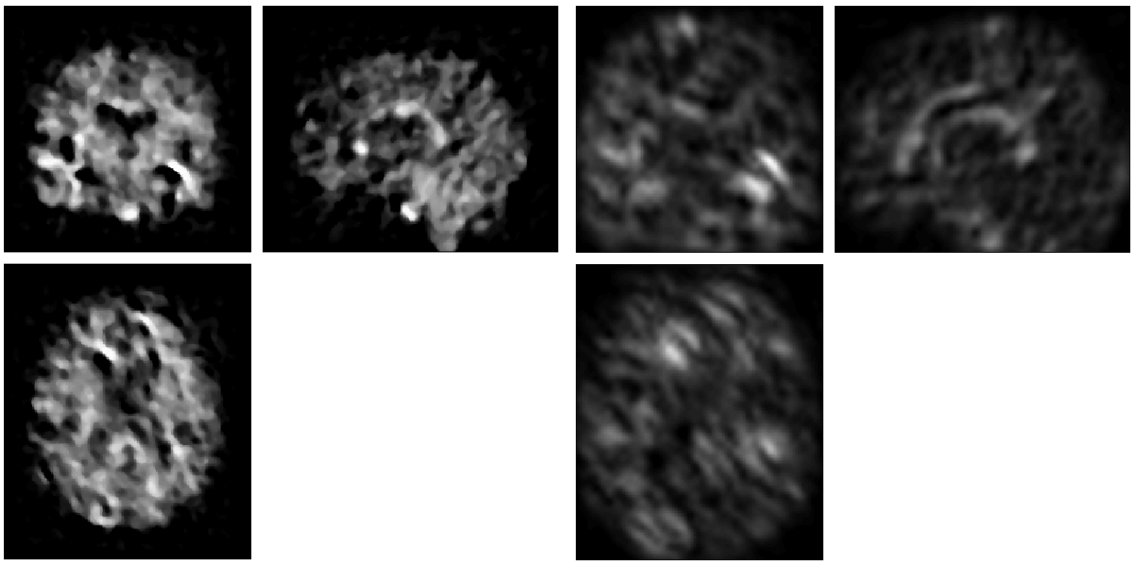
8.4. Results and Discussion

Fig. 8.2 shows the results of the CAMERA procedure. To begin, we will primarily look at Figs. 8.2a and 8.2c to analyze the activation maximization results on their own. While the



(a) Healthy activation maximization

(b) Healthy CAM



(c) AD activation maximization

(d) AD CAM

Figure 8.2. Healthy and AD activation maximizations and corresponding CAMs

boundaries of the brains are blurred as a result of the optimization, one can see a recession of the brain matter boundary in the AD visualization, potentially indicating the presence of atrophy associated with Alzheimer's disease. Furthermore, one can see enlargement of the ventricles in the center of the coronal view. One can even potentially make out increased

atrophy in the bottom right region of the coronal view, approximately in the location of the hippocampus. This is in line with biological phenomena that describes such atrophy in late stage Alzheimer's patients, especially around the hippocampus.

These results present some interpretable justification to the performance of the network; however, one could still make arguments that the network may not be paying attention to these regions in its classification. Thus, we turn to the CAM results in Figs. 8.2b and 8.2d. In these particular cases, the CAMs are larger than the original activation maximization brian volumes due to the rescaling that occurs during the CAM procedure in conjunction with the large receptive field at the last convolutional layer of the network. Regardless, one can compare the CAMs to the respective activation maximizations to see the relative network attention.

To begin, it seems that the network generally spreads its attention throughout the entire brain volume. This gives merit to our original decision to use the full 3D volume, as opposed to using slices or regions of interest as input to the CNN. Nonetheless, one can see some small regions of particular interest. In the coronal view, that is, the bottom right region of the top left sub-image, there is a hotspot in both the healthy and AD CAMs. This may provide justification for the network paying slightly more attention to the hippocampus region, which again falls in line with previously observed biological phenomena. One can also look at the axial view, specifically in the AD CAM, and see a small bright spot around the top left ventricle in the center of the brain. This could indicate that the network is looking at these ventricles as the first potential points of atrophy, which again falls in line with medical observations. Fundamentally, while these visualization do not prove without a doubt that the CNN is looking for features that are biologically interpretable to the medical field, the CAMERA procedure does gives credence to the notion.

8.5. Conclusion

We described CAMERA, a procedure that takes the output of activation maximization and passes it through Grad-CAM to yield particularly informative visualizations. We use the application of the 3D CNN used for binary AD classification and saw that, while the visualizations are not as crisp as the original brain volumes, the combination of activation maximizations and CAMs provided a deep look into the methodology behind the network. One could foresee using this CAMERA method in a variety of applications to improve the explainability of the two visualization schemes alone. For this particular application, one may look to comparing this CAMERA procedure to one performed on a graph CNN or capsule CNN applied to the same data to see the potential benefits of using more complex methodologies.

This dissertation has looked at multimodal data fusion and feature visualization in a variety of contexts. Chapter 2 outlined an investigation into applying CNNs to Alzheimer's classification using neuroimaging data, and Chapter 3 extended the investigation to cogntive regression and longitudinal prediction. Chapter 5 outlined the deep visualization technique of activation maximization and presented how it can be used to monitor CNN training. The power of activation maximization was also demonstrated in Chapters 6 and 7, where it was used in capsule networks and GANs. Finally, this chapter saw the combination of neuroimaging analysis with deep visualization. In the end, the benefits of this work can carry on past the confines of this dissertation, and hopefully can provide the genesis for other works going forward.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[2] Parnian Afshar, Arash Mohammadi, and Konstantinos N Plataniotis. Brain tumor type classification via capsule networks. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3129–3133. IEEE, 2018.

[3] Mordvintsev Alexander, C Olah, and Michael Tyka. Inceptionism: Going deeper into neural networks. *Google Research Blog*, 2015, 2015.

[4] Per-Arne Andersen. Deep reinforcement learning using capsules in advanced game environments. *arXiv preprint arXiv:1801.09597*, 2018.

[5] John Ashburner. A fast diffeomorphic image registration algorithm. *NeuroImage*, 38(1):95–113, 2007.

[6] Mohammad Taha Bahadori. Spectral capsule networks. 2018.

[7] DA Bennett, JA Schneider, Z Arvanitakis, JF Kelly, NT Aggarwal, RC Shah, and RS Wilson. Neuropathology of older persons without cognitive impairment from two community-based studies. *Neurology*, 66(12):1837–1844, 2006.

[8] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: A cpu and gpu math compiler in python. In *Proc. 9th Python in Science Conf*, volume 1, 2010.

[9] Nikhil Bhagwat, Jon Pipitone, Aristotle N Voineskos, M Mallar Chakravarty, Alzheimer's Disease Neuroimaging Initiative, et al. An artificial neural network model for clinical score prediction in alzheimer disease using structural neuroimaging measures. *Journal of psychiatry & neuroscience: JPN*, 44(4):246, 2019.

[10] Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the latent space of generative networks. *arXiv preprint arXiv:1707.05776*, 2017.

[11] Francois Chollet. How convolutional neural networks see the world. *The Keras Blog*, 30, 2016.

[12] François Chollet et al. Keras, 2015.

[13] Christopher M Clark, Lianne Sheppard, Gerda G Fillenbaum, Douglas Galasko, John C Morris, Elizabeth Koss, Richard Mohs, and Albert Heyman. Variability in annual mini-mental state examination score in patients with probable alzheimer disease: a clinical perspective of data from the consortium to establish a registry for alzheimer's disease. *Archives of neurology*, 56(7):857–862, 1999.

[14] Rémi Cuingnet, Emilie Gerardin, Jérôme Tessieras, Guillaume Auzias, Stéphane Lehéricy, Marie-Odile Habert, Marie Chupin, Habib Benali, Olivier Colliot, et al. Automatic classification of patients with alzheimer's disease from structural mri: a comparison of ten methods using the adni database. *NeuroImage*, 56(2):766–781, 2011.

[15] Fei Deng, Shengliang Pu, Xuehong Chen, Yusheng Shi, Ting Yuan, and Shengyan Pu. Hyperspectral image classification with capsule network using limited training samples. *Sensors*, 18(9):3153, 2018.

[16] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4829–4837, 2016.

[17] Kevin Duarte, Yogesh Rawat, and Mubarak Shah. Videocapsulenet: A simplified network for action detection. In *Advances in Neural Information Processing Systems*, pages 7610–7619, 2018.

[18] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[19] Bruce Fischl. Freesurfer. *Neuroimage*, 62(2):774–781, 2012.

[20] Bruce Fischl, Arthur Liu, and Anders M Dale. Automated manifold surgery: constructing geometrically accurate and topologically correct models of the human cerebral cortex. *IEEE transactions on medical imaging*, 20(1):70–80, 2001.

[21] Vladimir Fonov, Alan C Evans, Kelly Botteron, C Robert Almli, Robert C McKinstry, D Louis Collins, Brain Development Cooperative Group, et al. Unbiased average age-appropriate atlases for pediatric studies. *Neuroimage*, 54(1):313–327, 2011.

[22] Jon Gauthier. Conditional generative adversarial nets for convolutional face generation. *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester*, 2014(5):2, 2014.

[23] Emilie Gerardin, Gaël Chételat, Marie Chupin, Rémi Cuingnet, Béatrice Desgranges, Ho-Sung Kim, Marc Niethammer, Bruno Dubois, Stéphane Lehéricy, Line Garnero, et al. Multidimensional classification of hippocampal shape features discriminates alzheimer's disease and mild cognitive impairment from normal aging. *Neuroimage*, 47(4):1476–1486, 2009.

[24] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. 1999.

[25] Bangming Gong, Jun Shi, Shihui Ying, Yakang Dai, Qi Zhang, Yun Dong, Hedi An, and Yingchun Zhang. Neuroimaging-based diagnosis of parkinson's disease with deep neural mapping large margin distribution machine. *Neurocomputing*, 320:141–149, 2018.

[26] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[27] Brian A Gordon, Tyler M Blazey, Yi Su, Amrita Hari-Raj, Aylin Dincer, Shaney Flores, Jon Christensen, Eric McDade, Guoqiao Wang, Chengjie Xiong, et al. Spatial patterns of neuroimaging biomarker change in individuals from families with autosomal dominant alzheimer's disease: a longitudinal study. *The Lancet Neurology*, 17(3):241–250, 2018.

[28] Jingtao Guo, Zhenzhen Qian, Zuowei Zhou, and Yi Liu. Mulgan: Facial attribute editing by exemplar. *arXiv preprint arXiv:1912.12396*, 2019.

[29] Ashish Gupta, Murat Ayhan, and Anthony Maida. Natural image bases to represent neuroimaging data. In *International Conference on Machine Learning*, pages 987–994, 2013.

[30] Richard HR Hahnloser, Rahul Sarpeshkar, Misha A Mahowald, Rodney J Douglas, and H Sebastian Seung. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*, 405(6789):947–951, 2000.

[31] LE Hebert, PA Scherr, JL Bienias, DA Bennett, and DA Evans. State-specific projections through 2025 of alzheimer disease prevalence. *Neurology*, 62(9):1645–1645, 2004.

[32] Geoffrey E Hinton, Sara Sabour, and Nicholas Frosst. Matrix capsules with em routing. 2018.

[33] Geoffrey E Hinton and Richard S Zemel. Autoencoders, minimum description length and helmholtz free energy. In *Advances in neural information processing systems*, pages 3–10, 1994.

[34] Ehsan Hosseini-Asl, Robert Keynton, and Ayman El-Baz. Alzheimer's disease diagnostics by adaptation of 3d convolutional network. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 126–130. IEEE, 2016.

[35] Tomas Iesmantas and Robertas Alzbutas. Convolutional capsule network for classification of breast cancer histology images. In *International Conference Image Analysis and Recognition*, pages 853–860. Springer, 2018.

[36] Kazi Aminul Islam, Daniel Pérez, Victoria Hill, Blake Schaeffer, Richard Zimmerman, and Jiang Li. Seagrass detection in coastal water through deep capsule networks. In *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, pages 320–331. Springer, 2018.

[37] Clifford R Jack, David A Bennett, Kaj Blennow, Maria C Carrillo, Billy Dunn, Samantha Budd Haeberlein, David M Holtzman, William Jagust, Frank Jessen, Jason Karlawish, et al. Nia-aa research framework: Toward a biological definition of alzheimer's disease. *Alzheimer's & Dementia*, 14(4):535–562, 2018.

[38] Clifford R Jack, Matt A Bernstein, Nick C Fox, Paul Thompson, Gene Alexander, Danielle Harvey, Bret Borowski, Paula J Britson, Jennifer L Whitwell, Chadwick Ward, et al. The alzheimer's disease neuroimaging initiative (adni): Mri methods. *Journal of magnetic resonance imaging*, 27(4):685–691, 2008.

[39] Clifford R Jack Jr, David S Knopman, William J Jagust, Ronald C Petersen, Michael W Weiner, Paul S Aisen, Leslie M Shaw, Prashanthi Vemuri, Heather J Wiste, Stephen D Weigand, et al. Tracking pathophysiological processes in alzheimer's disease: an updated hypothetical model of dynamic biomarkers. *The Lancet Neurology*, 12(2):207–216, 2013.

[40] Ayush Jaiswal, Wael AbdAlmageed, Yue Wu, and Premkumar Natarajan. Capsulegan: Generative adversarial capsule network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[41] Mark Jenkinson, Peter Bannister, Michael Brady, and Stephen Smith. Improved optimization for the robust and accurate linear registration and motion correction of brain images. *Neuroimage*, 17(2):825–841, 2002.

[42] Amelia Jiménez-Sánchez, Shadi Albarqouni, and Diana Mateus. Capsule networks against medical imaging data challenges. In *Intravascular Imaging and Computer Assisted Stenting and Large-Scale Annotation of Biomedical Data and Expert Label Synthesis*, pages 150–160. Springer, 2018.

[43] Aggelos K Katsaggelos, Sara Bahaadini, and Rafael Molina. Audiovisual fusion: Challenges and new approaches. *Proceedings of the IEEE*, 103(9):1635–1653, 2015.

[44] Youngjoo Kim, Peng Wang, Yifei Zhu, and Lyudmila Mihaylova. A capsule network for traffic speed prediction in complex road networks. In *2018 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, pages 1–6. IEEE, 2018.

[45] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[46] Stefan Klöppel, Cynthia M Stonnington, Carlton Chu, Bogdan Draganski, Rachael I Scahill, Jonathan D Rohrer, Nick C Fox, Clifford R Jack Jr, John Ashburner, and Richard SJ Frackowiak. Automatic classification of mr scans in alzheimer's disease. *Brain*, 131(3):681–689, 2008.

[47] Amara Dinesh Kumar. Novel deep learning model for traffic sign detection using capsule networks. *arXiv preprint arXiv:1805.04424*, 2018.

[48] Dana Lahat, Tülay Adali, and Christian Jutten. Multimodal data fusion: an overview of methods, challenges, and prospects. *Proceedings of the IEEE*, 103(9):1449–1477, 2015.

[49] Rodney LaLonde and Ulas Bagci. Capsules for object segmentation. *arXiv preprint arXiv:1804.04241*, 2018.

[50] Susan M Landau, Mark A Mintun, Abhinay D Joshi, Robert A Koeppe, Ronald C Petersen, Paul S Aisen, Michael W Weiner, William J Jagust, and Alzheimer's Disease Neuroimaging Initiative. Amyloid deposition, hypometabolism, and longitudinal cognitive decline. *Annals of neurology*, 72(4):578–586, 2012.

[51] Y LeCun, Y Bengio, and G Hinton. Deep learning. nature 521 (7553): 436. *Google Scholar*, 2015.

[52] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[53] Feng Li, Loc Tran, Kim-Han Thung, Shuiwang Ji, Dinggang Shen, and Jiang Li. A robust deep model for improved classification of ad/mci patients. *IEEE journal of biomedical and health informatics*, 19(5):1610–1616, 2015.

[54] Rongjian Li, Wenlu Zhang, Heung-Il Suk, Li Wang, Jiang Li, Dinggang Shen, and Shuiwang Ji. Deep learning based imaging data completion for improved brain disease diagnosis. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 305–312. Springer, 2014.

[55] Yu Li, Meiyu Qian, Pengfeng Liu, Qian Cai, Xiaoying Li, Junwen Guo, Huan Yan, Fengyuan Yu, Kun Yuan, Juan Yu, et al. The recognition of rice images by uav based on capsule network. *Cluster Computing*, pages 1–10, 2018.

[56] Zachary C Lipton and Subarna Tripathi. Precise recovery of latent vectors from generative adversarial networks. *arXiv preprint arXiv:1702.04782*, 2017.

[57] Mingxia Liu, Daoqiang Zhang, Ehsan Adeli, and Dinggang Shen. Inherent structure-based multiview learning with multitemplate feature representation for alzheimer's disease diagnosis. *IEEE Transactions on Biomedical Engineering*, 63(7):1473–1482, 2016.

[58] Siqi Liu, Sidong Liu, Weidong Cai, Hangyu Che, Sonia Pujol, Ron Kikinis, Dagan Feng, Michael J Fulham, et al. Multimodal neuroimaging feature learning for multiclass diagnosis of alzheimer's disease. *IEEE Transactions on Biomedical Engineering*, 62(4):1132–1140, 2015.

[59] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5188–5196, 2015.

[60] Aravindh Mahendran and Andrea Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision*, 120(3):233–255, 2016.

[61] Harry McGurk and John MacDonald. Hearing lips and seeing voices. *Nature*, 264(5588):746, 1976.

[62] ER McVeigh, MJ Bronskill, and RM Henkelman. Phase and sensitivity of receiver coils in magnetic resonance imaging. *Medical physics*, 13(6):806–814, 1986.

[63] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[64] Chandan Misra, Yong Fan, and Christos Davatzikos. Baseline and longitudinal patterns of brain atrophy in mci patients, and their use in prediction of short-term conversion to ad: results from adni. *Neuroimage*, 44(4):1415–1422, 2009.

[65] Aryan Mobiny and Hien Van Nguyen. Fast capsnet for lung cancer screening. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 741–749. Springer, 2018.

[66] Sudipto Mukherjee, Himanshu Asnani, Eugene Lin, and Sreeram Kannan. Clustergan: Latent space clustering in generative adversarial networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4610–4617, 2019.

[67] James O' Neill. Siamese capsule networks. *arXiv preprint arXiv:1805.07242*, 2018.

[68] Peter T Nelson, Elizabeth Head, Frederick A Schmitt, Paulina R Davis, Janna H Neltner, Gregory A Jicha, Erin L Abner, Charles D Smith, Linda J Van Eldik, Richard J Kryscio, et al. Alzheimer's disease is not "brain aging": neuropathological, genetic, and epidemiological human studies. *Acta neuropathologica*, 121(5):571–587, 2011.

[69] Anh Nguyen, Jason Yosinski, and Jeff Clune. Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *arXiv preprint arXiv:1602.03616*, 2016.

[70] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2642–2651. JMLR. org, 2017.

[71] Adrien Payan and Giovanni Montana. Predicting alzheimer's disease: a neuroimaging study with 3d convolutional neural networks. *arXiv preprint arXiv:1502.02506*, 2015.

[72] Joseph L Price, PB Davis, JC Morris, and DL White. The distribution of tangles, plaques and related immunohistochemical markers in healthy aging and alzheimer's disease. *Neurobiology of aging*, 12(4):295–312, 1991.

[73] Arjun Punjabi and Aggelos K Katsaggelos. Visualization of feature evolution during convolutional neural network training. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 311–315. IEEE, 2017.

[74] David Rawlinson, Abdelrahman Ahmed, and Gideon Kowadlo. Sparse unsupervised capsules generalize better. *arXiv preprint arXiv:1804.06094*, 2018.

[75] Hao Ren and Hong Lu. Compositional coding capsule network with k-means routing for text classification. *arXiv preprint arXiv:1810.09177*, 2018.

[76] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[77] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866, 2017.

[78] Mert R Sabuncu, Ender Konukoglu, Alzheimer's Disease Neuroimaging Initiative, et al. Clinical prediction from structural brain mri scans: a large-scale empirical study. *Neuroinformatics*, 13(1):31–46, 2015.

[79] Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 28(11):2660–2673, 2017.

[80] Yassir Saquil, Kwang In Kim, and Peter Hall. Ranking cgans: Subjective control over semantic image attributes. *arXiv preprint arXiv:1804.04082*, 2018.

[81] Saman Sarraf and Ghassem Tofighi. Classification of alzheimer's disease using fmri data and deep learning convolutional neural networks. *arXiv preprint arXiv:1603.08631*, 2016.

[82] Saman Sarraf, Ghassem Tofighi, et al. Deepad: Alzheimer's disease classification via deep convolutional neural networks using mri and fmri. *bioRxiv*, page 070441, 2016.

[83] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[84] Atefeh Shahroudnejad, Parnian Afshar, Konstantinos N Plataniotis, and Arash Mohammadi. Improved explainability of capsule networks: Relevance path by agreement. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 549–553. IEEE, 2018.

[85] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. *arXiv preprint arXiv:1907.10786*, 2019.

[86] Jun Shi, Zeyu Xue, Yakang Dai, Bo Peng, Yun Dong, Qi Zhang, and Yingchun Zhang. Cascaded multi-column rvfl+ classifier for single-modal neuroimaging-based diagnosis of parkinson's disease. *IEEE Transactions on Biomedical Engineering*, 2018.

[87] Jun Shi, Xiao Zheng, Yan Li, Qi Zhang, and Shihui Ying. Multimodal neuroimaging feature learning with multimodal stacked deep polynomial networks for diagnosis of alzheimer's disease. *IEEE journal of biomedical and health informatics*, 22(1):173–183, 2018.

[88] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[89] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[90] John G Sled, Alex P Zijdenbos, and Alan C Evans. A nonparametric method for automatic correction of intensity nonuniformity in mri data. *IEEE transactions on medical imaging*, 17(1):87–97, 1998.

[91] Stephen M Smith. Fast robust automated brain extraction. *Human brain mapping*, 17(3):143–155, 2002.

[92] Cees GM Snoek, Marcel Worring, and Arnold WM Smeulders. Early versus late fusion in semantic video analysis. In *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 399–402. ACM, 2005.

[93] Heung-Il Suk, Seong-Whan Lee, Dinggang Shen, Alzheimer's Disease Neuroimaging Initiative, et al. Hierarchical feature representation and multimodal fusion with deep learning for ad/mci diagnosis. *NeuroImage*, 101:569–582, 2014.

[94] Heung-Il Suk and Dinggang Shen. Deep learning-based feature representation for ad/mci classification. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 583–590. Springer, 2013.

[95] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[96] Solale Tabarestani, Maryamossadat Aghili, Mehdi Shojaie, Christian Freytes, Mercedes Cabrerizo, Armando Barreto, Naphtali Rishe, Rosie E Curiel, David Loewenstein, Ranjan Duara, et al. Longitudinal prediction modeling of alzheimer disease using recurrent neural networks. In *2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, pages 1–4. IEEE, 2019.

[97] Reeti Tandon, Sudeshna Adak, and Jeffrey A Kaye. Neural networks for longitudinal studies in alzheimer's disease. *Artificial intelligence in medicine*, 36(3):245–255, 2006.

[98] Fabian Tence. Visualizing deep neural networks classes and features, Jul 2016.

[99] Nicholas J Tustison, Brian B Avants, Philip A Cook, Yuanjie Zheng, Alexander Egan, Paul A Yushkevich, and James C Gee. N4itk: improved n3 bias correction. *IEEE transactions on medical imaging*, 29(6):1310–1320, 2010.

[100] Andrey Voynov and Artem Babenko. Unsupervised discovery of interpretable directions in the gan latent space. *arXiv preprint arXiv:2002.03754*, 2020.

[101] Tien Duong Vu, Hyung-Jeong Yang, Van Quan Nguyen, A-Ran Oh, and Mi-Sun Kim. Multimodal learning using convolution neural network and sparse autoencoder. In *Big Data and Smart Computing (BigComp), 2017 IEEE International Conference on*, pages 309–312. IEEE, 2017.

[102] Jing Wan, Zhilin Zhang, Jingwen Yan, Taiyong Li, Bhaskar D Rao, Shiaofen Fang, Sungeun Kim, Shannon L Risacher, Andrew J Saykin, and Li Shen. Sparse bayesian multi-task learning for predicting cognitive outcomes from neuroimaging measures in alzheimer's disease. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 940–947. IEEE, 2012.

[103] Yequan Wang, Aixin Sun, Jialong Han, Ying Liu, and Xiaoyan Zhu. Sentiment analysis by capsules. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 1165–1174. International World Wide Web Conferences Steering Committee, 2018.

[104] Edgar Xi, Selina Bing, and Yang Jin. Capsule network performance on complex data. *arXiv preprint arXiv:1712.03480*, 2017.

[105] Canqun Xiang, Lu Zhang, Yi Tang, Wenbin Zou, and Chen Xu. Ms-capsnet: A novel multi-scale capsule network. *IEEE Signal Processing Letters*, 25(12):1850–1854, 2018.

[106] Xin Yang, Qiang Wu, Don Hong, and Jiancheng Zou. Spatial regularization for neural network and application in alzheimer's disease classification. In *Future Technologies Conference (FTC)*, pages 831–837. IEEE, 2016.

[107] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[108] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[109] Guan Yu, Yufeng Liu, and Dinggang Shen. Graph-guided joint prediction of class label and clinical scores for the alzheimer's disease. *Brain Structure and Function*, 221(7):3787–3801, 2016.

[110] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[111] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[112] Daoqiang Zhang, Dinggang Shen, Alzheimer's Disease Neuroimaging Initiative, et al. Multi-modal multi-task learning for joint prediction of multiple regression and classification variables in alzheimer's disease. *NeuroImage*, 59(2):895–907, 2012.

[113] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014.

[114] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.

[115] Fan Zhu, Bharat Panwar, Hiroko H Dodge, Hongdong Li, Benjamin M Hampstead, Roger L Albin, Henry L Paulson, and Yuanfang Guan. Compass: A computational model to predict changes in mmse scores 24-months after initial assessment of alzheimer's disease. *Scientific reports*, 6:34567, 2016.

[116] Xiaofeng Zhu, Heung-Il Suk, and Dinggang Shen. A novel matrix-similarity based loss function for joint regression and classification in ad diagnosis. *NeuroImage*, 100:91–105, 2014.

[117] Luisa M Zintgraf, Taco S Cohen, and Max Welling. A new method to visualize deep neural networks. *arXiv preprint arXiv:1603.02518*, 2016.

[118] Chen Zu, Biao Jie, Mingxia Liu, Songcan Chen, Dinggang Shen, Daogiang Zhang, Alzheimer's Disease Neuroimaging Initiative, et al. Label-aligned multi-task feature learning for multimodal classification of alzheimer's disease and mild cognitive impairment. *Brain imaging and behavior*, 10(4):1148–1159, 2016.