Journal of High School Science

An investigation of Semi-Supervised Learning with autoencoders to improve high dimensional image classification with fewer labels

Vitali F[1], Showalter S[2]

Submitted: November 12, 2023, Revised: version 1, March 16, 2024, version 2, March 17, 2024
Accepted: March 17, 2024

Abstract

As the field of machine learning expands, so does the need to learn patterns in high-dimensional image data, a problem we explore with semi-supervised techniques. This learning becomes problematic through the Curse of Dimensionality which explains that as data dimensionality increases, performance decreases. Semi-supervised learning has been proposed as a solution to this problem, by leveraging a small portion of labeled data and a larger unlabeled set. We sought to explore model performance under various model sizes, and the percentage of labeled data. We found that in small amounts of labeled data, the moderately sized model performed best. As the labeled data percentage increases, the larger model obtains the higher test accuracy. Finally, the learning patterns of each of these models are "S"-curves, representing a 'sweet spot' of linear semi-supervised learning on a moderate amount of labeled data.

Keywords

Semi-Supervised Learning, Autoencoders, High-dimensional images, Information bottleneck, MNIST, Curse of dimensionality, Deep learning, Model complexity, Unlabeled data, Fewer Labels, Supervision ratio

[1]Corresponding author: Fiona Vitali, Bernards High School, 25 Olcott Ave, Bernardsville, NJ 07924, USA. fiona.m.vitali@gmail.com
[2]Samuel Showalter, UC Irvine Donald Bren School of Information and Computer Sciences, Irvine, CA 92692, USA. showalte@uci.edu

Introduction

Deep learning is characterized by learning the parameters of a neural network such that it can identify statistical patterns of interest. These machine learning methods are particularly powerful because neural networks have been shown to be universal function approximators that can learn any function given enough data and capacity (1). Neural networks have been tailored to learn patterns from innumerable data modalities, including from image data. This represents significant progress of intelligent automation, as image and video data represent nearly 80% of data on the internet by memory (2).

However, learning from data is challenging when input dimensionality is large, with images being especially cumbersome given their 3D (height, width, color channels) composition. Generally, the semantics of an image can be represented with fewer dimensions than the number of pixels, because background pixels may carry little information. When learning methods are applied naively in this setting, we encounter the curse of dimensionality or the Hughes phenomenon (3). This phenomenon states that as the number of features in the data increases, the performance increases until the feature dimension is comparable to the number of observations, and then subsequently decreases, thereby leading to a decrease in performance due to data sparsity. In this regime, models begin to memorize training data rather than learn a generalizable decision boundary. This is the classical condition known as overfitting. Due to the high dimensionality of images, this negative pathology is often seen when learning is done in image space. We investigated the potential to avoid these negative outcomes in low data and model capacity settings using Semi-Supervised Learning.

Dataset size must grow exponentially to combat the curse of dimensionality (3). However, such a large volume of data is typically intractable to collect or too expensive to individually label. To combat this common challenge in machine learning, Semi-Supervised Learning (SSL) was proposed as a solution. Under SSL, only a portion of the total data is labeled and the model is concurrently trained on unsupervised losses over this larger unlabeled dataset (4). There continues to be standard splitting of the data, but the question is how much of the training set is labeled and how much of it is unlabeled. In situations where labeling large amounts of data is time-consuming, Semi-Supervised Learning seeks better generalization from a small amount of labeled data by leveraging patterns in the unlabeled samples (5). We explored the potential of SSL in the setting of handwritten digit classification - a high-dimensional image problem. In addition, we also examined the less explored interaction effects of model capacity and labeled data quantity on the effectiveness of SSL. That is, we asked the question, will unlabeled data always be beneficial to include in high dimensional problems, or are there sub-cases where it is instead detrimental?

Semi-Supervised Learning combines supervised and unsupervised objectives. A common unsupervised objective is the autoencoder loss, which compresses high-dimensional input and then reconstructs the

input again from the compressed latent representation. This compression serves as an additional means of combating the curse of dimensionality by use of the Information Bottleneck principle. Using autoencoders in conjunction with SSL could potentially improve the predictive performance of Machine Language (ML) models in data-scarce settings.

Previous experiments with hyperparameter selection have found an optimal point in the data volume where the performance improvement with additional data became marginal beyond a certain point. Zhang et al. found that this optimal threshold occurred at 10-20% of their total data (6). We expected a similar, but hopefully lower, percentage using our SSL technique and use of autoencoders. We explored SSL performance under various levels of supervision and model capacity in conjunction with autoencoder dimensionality reduction and reconstruction objectives, aiming to improve multi-class prediction on high-dimensional images.

Literature review

Within the realm of semi-supervised learning, there are several approaches that have been explored. Below, we discuss the main families of SSL and provide an overview of their characteristics and applications, closing with a highlight of our intended approach and how it fits into this greater landscape.

One prominent family of SSL is representation learning, a sub-domain of ML that trains an algorithm to discover compact latent representations that may be interpretable and/or utilized for transfer learning, etc (7). Autoencoders, a neural network used to reconstruct text and images, have emerged as a popular choice for representation learning within SSL. Considered alone, the autoencoder training objective is completely self-supervised. Autoencoders encode input data into a lower-dimensional representation before reconstructing the original input from this compressed latent, leveraging the bottleneck Method from Information Theory, which in turn trades off accuracy with compression (8).

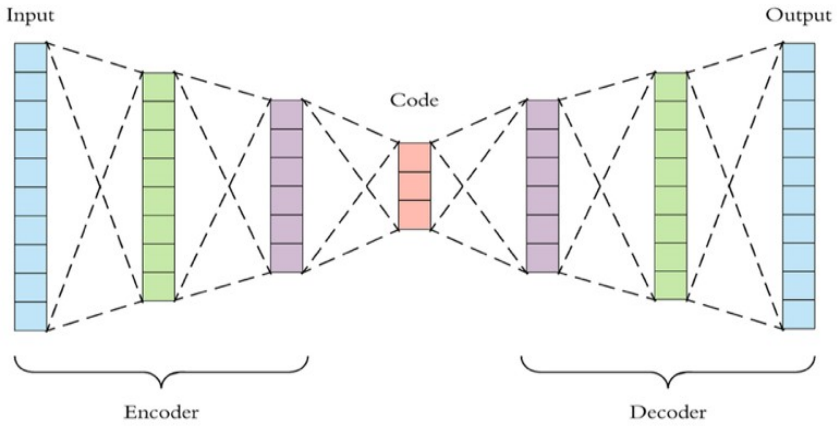

Figure 1.1. Visual Representation of Autoencoders (16)

In learning a compressed representation of the semantic features of the image data, autoencoders can assist in training models for classification tasks on high-dimensional data. Along with standard autoencoders, masked autoencoders are another effective technique for learning a compressed representation of input data which maintains its semantic structure. These autoencoders remove a portion of the input image and are then trained to reconstruct the missing information. Masked autoencoders, though immensely popular, represent only a single approach to the broader application of denoising autoencoding, which can recover corrupted data samples (9).

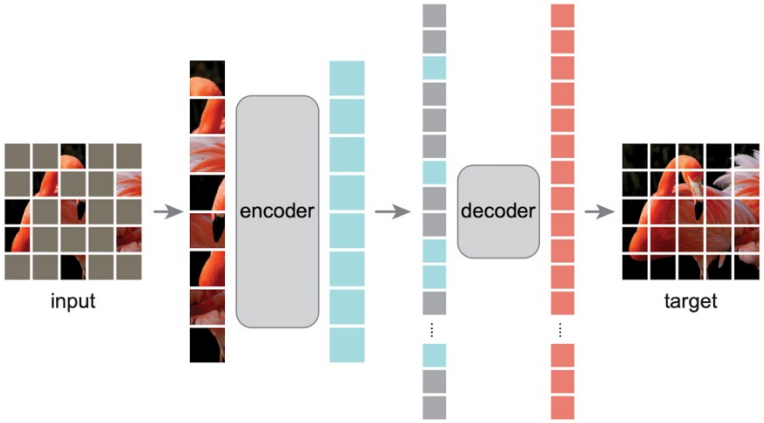

Figure 1.2. Visual Representation of Masked Autoencoders (8)

Capturing meaningful relationships in the data assists in dimensionality reduction and denoising. Most importantly, the masking process offers a strong self-supervision signal with which these models can learn the data's underlying structure. Training the model to fill in masked sections of data encodes much of the essential information about the image in a compressed manner that relates different inputs to one another and may even disentangle features (10). Learning compact representations expands the masked autoencoder's use into natural language processing, computer vision, and recommendation system domains.

Beyond autoencoders and infilling, pretext tasks such as self-supervised rotation prediction have achieved valuable improvements over traditional training. Incorporating this auxiliary task in representation learning allows the model to capture the input data orientation or learn valuable invariances, thereby enabling the model to learn robust discriminative features. The use of both representation learning techniques of autoencoders and rotation prediction enhances the accuracy of ML models while minimizing or negating the need for extensive labeled data. Despite their prevalence, all of these aforementioned approaches depend on input space

augmentations. Below, we discuss SSL methods that rely on augmentations of the target space.

One example of target-space augmentation is pseudo-labeling, a technique that uses high-confidence predicted labels to bootstrap learning. The main principle of pseudo-labeling is using an initialized model to noisily assign pseudo labels on the unlabeled data given very few samples. The model then trains using augmented samples and attempts to match these generated pseudo labels to refine its predictions (11). This self-training process improves the model's performance by learning discriminative features in a self-supervised manner. A modern pseudo-labeling technique is FixMatch, which creates pseudo labels on weakly-augmented unlabeled images with high confidence before training on the strongly-augmented version of the same image. The model improves its generalization capability by producing similar outputs for the augmented versions of the same input. Extending the concept of FixMatch, FlexMatch incorporates dynamic confidence thresholds. This represents a method of contrastive learning that is applied to the label space rather than to the feature space. In contrast with a fixed threshold, these thresholds adjust based on the model's performance per class to increase robustness and flexibility in noisy data (12). Both FixMatch and FlexMatch in contrastive learning are effective self-training techniques that enhance the model's ability to generalize and achieve higher SSL accuracies.

Another related family of SSL for learning representations is contrastive learning which learns by instance discrimination: matching up semantically alike samples with noisy or differing features. This technique is similar to pretext task learning but extends the concepts to include different samples. It should be noted that while pseudo-labeling also does this, it does so in the label space. This can also be formulated as an alignment of two corruptions of the same sample in an embedding space to capture the data's structure. Momentum Contrast (MOCO) and SimCLR are two methods that employ this SSL instance discrimination technique, also known as representation alignment. Both techniques use various augmentations, such as random cropping, color distortion, and Gaussian blurring, which are used on each augmented view of the input samples. MOCO is a SSL algorithm with a contrastive loss. It achieves this by training a momentum encoder to a matched encoded query to a dictionary of encoded keys from contrastive loss, negating the need for a large batch size. The momentum encoder's parameters are updated throughout training but are delayed compared to the primary encoder (13). This allows for more stable representations. SimCLR, however, uses a similar SSL framework by maximizing the different augmented views of the same instance through a contrastive loss in the latent space without a queue or momentum encoder. The model is then trained on these different augmented views into representations in the embedding space.

This literature review on SSL branches and techniques established the foundation for our research objective; enhancing our ability to learn models on high-dimensional image data

with fewer labels. SSL combines the strengths proposed by both supervised and unsupervised learning to lower labeling costs and is improved by a semantically compact representation (14). Autoencoders, a frequent choice in SSL models, encode input data, compress it, and effectively reduce the dimensionality. Integration of these representation learning techniques aligned with our objective of improving the interpretation accuracy of high-dimensional data through interpretable representations. The evolution of a variety of SSL techniques, such as the ones stated previously, have led to significant advancements and further experimentation.

Methods

Our exploration in classifying high-dimensional data with fewer labels focused on comparing the generalization capabilities of machine learning models under different quantities of labeled data, leveraging the remainder in an unsupervised fashion. To maintain fairness in comparison, we utilized a single dataset for our experiments but modulated the level of supervision. In doing so, we ensured that changes in accuracy were attributed to the levels of supervision and not to any other factors.

In turn, we compared the accuracy percentages achieved with decreasing quantity of labeled data on images with autoencoders. To regularize our model and prevent overfitting, we leveraged a semi-supervised learning scheme. Unlabeled data was fed to an autoencoder, and a self-supervised model was built for representation learning. The autoencoders learned valuable inductive biases without labels, and we hoped that this would improve model generalization. This increased accuracy even with a decreased quantity of labeled data would then allow for training the machine learning models on high-dimensional data, easier and less expensive.

With this viewpoint of semi-supervised learning, we formulated our classification problem to learn a multi-task model. We began with an encoder denoted as $z = h(x|\theta)$ which produced a latent representation. Subsequently, this latent was passed to both a classification head $\hat{y} := f(z|\varphi)$ and decoder $\hat{x} := g(z|\psi)$, where θ, φ, ψ defined the learnable parameters. We note that all parameters are learned end-to-end, but differentiate them here for clarity of purpose. Taken together, the model approximated the probability distribution $p(y|x)$ using a labeled dataset D=(x,y) and the distribution p(x) of unlabeled dataset $U = (x_u)$. Here, x represents an image with dimensions $R^{(h \times w \times c)}$, where h, w, and c represent height, width, and number of RGB channels. The corresponding label for each image is denoted by y. To incorporate semi-supervised learning techniques, we augmented our classifier f with an autoencoder head, represented as $AE(z, \theta)$. The autoencoder mapped an input image x to a lower-dimensional latent representation, $z \in R^{(h \times w \times c)}$. The autoencoder's reconstructed image is denoted as x_r. These formulas represent the underlying structure of the algorithm on which we built our model.

In order to track the level of supervision for each experiment, we defined the supervision ratio as a. This represented the percentage of labeled images in the combined dataset

D ∪ U. During our experiments, we considered various values of a to investigate the effect of supervision on the performance of the SSL with autoencoders. $f(x)$. The image classifier was initially trained using the SSL, where the loss function incorporated both labeled data and the unsupervised reconstruction loss from the autoencoder. The supervised cross-entropy loss measured the difference between the predicted probabilities and the true labels. For a labeled x_i image with y_i label, the loss function was defined as $L_{CE}(D) = -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K}\sum_{j=1}^{K} y_i \log(\hat{p}\,(y_j|\,x_i))$ where N is the number of samples, K is the number of classes, and $\hat{y} = f \circ h(x)$. The unsupervised reconstruction loss used the mean squared error (MSE), to evaluate the quality of the autoencoder's image reconstruction. For an image x_i, the MSE loss is $L_{MSE}(x_j) = \frac{1}{d}\sum_{j=1}^{d}(x_{i,j} - g \circ h(x_i)_j)^2$ and computed over all samples. By employing these notations for the loss functions, we defined and evaluated the performance of our SSL model with autoencoders on high-dimensional image data reduction.

To perform these experiments, we used the MNIST dataset, which contains images of handwritten digits with corresponding labels indicating the digit each image represents. The MNIST dataset is sufficiently complex for investigating an SSL, and has been validated in many SOTA papers in the literature. Each supervision level was determined by the supervision level percentage as mentioned previously and using the conditions of 'only labeled' (L) or a combination of 'unlabeled and labeled' (U+L) data. The results of each experiment were also evaluated using three linear models: large, medium, and small. The large linear model used 1,152,314 parameters, and represented a complex model with a high level of model capacity. The medium linear model used 210,298 parameters, and represented a relatively large model that would best be suitable for simpler tasks. Lastly, the small linear model incorporated 52,234 parameters with significantly fewer parameters than the other models and was more suitable for lightweight tasks. These model sizes allowed us to analyze how the labeled data percentage and condition of unlabeled data affected the accuracy achieved for a given model size.

These models were hosted on Google Colaboratory in Python, using the Pytorch library. This was our main setup for the research and allowed us free access to accelerated hardware. To begin the research, we downloaded the MNIST dataset to Google Colab. Next, we created a specialized data loader, based on the PyTorch custom data loader, with adjustable supervision. This flexibility of supervision allowed us to manage the number of images; labeled or unlabeled; for our tests. Once we downloaded our test and the labeled and unlabeled training data, we executed the neural network with five layers. The first, third, and fifth layers were linear and the second and fourth layers were the Rectified Linear Unit (ReLu) activation function denoted as $f(x) = \max(0,x)$.

We used PyTorch to implement the model and used its backpropagation capabilities to efficiently train it. By tuning the parameters of

the network—batch size, labeled data percentage, maximum training iterations, learning rate, and checkpoint iterations—we achieved a well-performing image classifier that generalized effectively on the unseen data. In running our experiments, we tested with 0.01%, 0.1%, 1%, 5%, 7%, 10%, 15%, 20%, and 50% labeled data. We tested three data configurations; 'only labeled' data (L), 'only unlabeled' data (U) and a combination of 'labeled and unlabeled' data (U+L), on each of these tests with different percentages of labeled and unlabeled data. Any difference in performance between labeled data percentages greater than 10% was considered significant. Our evaluation model ran the data through the model, switched to the GPU for faster computation power, calculated predictions and accuracies, and returned those accuracies. Using this evaluation function helped track our performance and identify how each experiment affected the overall accuracy.

Results

In this section, we presented the results of our experiments and their consequential analysis. As seen in the graph below of performance across all levels of labeled data, the first 100-150 iterations constantly changed but as the model reached close to 400 iterations, the accuracies predominantly flattened out at their respective accuracy levels.

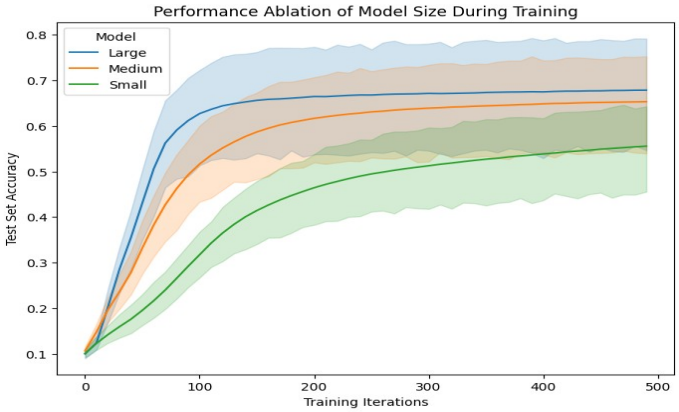

Figure 1.3. Graph for Performance Ablation of Model Size During Training

The detailed evaluation of each model and experiment with their respective test accuracies are shown in Table 1. For the 0.01% test, the medium linear model (L) pressented with the highest accuracy at 16.58%. At the same percentage of labeled data (0.01%), the small linear model (U+L) presented with a higher accuracy (14.19%) than the large linear model (U+L) of 10.09%. At this labeled data percentage, the medium linear model trained the highest accuracy, in both U+L and L data, among all the three models. This suggested that a moderately complex model was more effective at utilizing the limited labeled data and leveraging the unlabeled data at this labeled data percentage. This observation changed throughout the tests. For the labeled data, the medium linear model (L) was

significantly more accurate at 16.58% compared to the large model (L) and small linear (L) model's 9.74% and 9.56%---meaning that yet again, the medium linear model was able to extract more discriminative information, only this time from the limited labeled data.

Table 1. Accuracies of the three models as a function of percent labeled data.

Percent labeled data	Large Model		Medium Model		Small Model	
	U + L	L	U + L	L	U + L	L
0.01%	**10.09**	09.74	15.47	**16.58**	**14.19**	09.56
0.1%	29.92	**37.23**	30.25	**40.36**	**27.86**	24.02
1%	**68.55**	61.02	**65.92**	51.26	**59.99**	54.69
5%	**82.23**	76.08	78.09	**82.02**	**69.79**	60.48
7%	**79.78**	77.37	**76.06**	74.99	73.68	**75.61**
10%	74.77	**86.75**	82.83	**83.15**	**69.31**	60.04
15%	82.95	**88.71**	72.38	**80.27**	68.07	**70.85**
20%	84.91	**89.15**	77.49	**79.89**	69.77	**71.56**
50%	89.80	**91.79**	**84.39**	83.88	56.21	**64.13**

The 0.1% test saw significant increases in test accuracies for every model, both for U+L as well as for L. Similar to the test at 0.01%, the medium linear model presented with the highest test accuracies compared to the other models, both with and without unlabeled data at 30.25% and 40.35% when compared with their corresponding counterparts. The large linear model (L) yielded the largest increase in percentage accuracy from its corresponding value for the 0.01% test; increasing 3.82 fold from 9.74% to 37.23%. This significant increase in accuracy in the large linear model (L) going from 0.01% to 0.1% labeled data were also seen; albeit not to this extent; in the other models. The large, medium and small U+L models increased in accuracy 2.96, 1.96 and 1.96 fold respectively, while the medium and small L models increase in accuracy 2.43 and 2.51 fold respectively.

At 1% labeled data, the large linear model, both at (U+L) and L, presented with the highest test accuracies at 68.55% and 61.02% respectively, compared with the accuracies of the corresponding (U+L) and L medium and small models. At 1% labeled data, these accuracies of the large linear model were significantly greater than those of the medium and small models, except when comparing the accuracies for the (U+L) large and medium models, where the difference was not

significant. Although the medium linear model (L) presented with the least accuracy at 51.26%, when compared against its counterpart, it did reverse this trend in the next test at 5% labeled data. The small (U+L) model accuracy increased the least from the 0.1% labeled data at 2.15 fold, versus 2.18 and 2.29 fold for the medium and large (U+L) models respectively. The small L model's accuracy however increased the most from the 0.1% labeled data at 2.28 fold, versus 1.27 and 1.63 fold for the medium and large (L) models respectively. The rate of increase for the small model would however begin to decrease across the board (for both the U+L and L models) with further increase in the percent labeled data (discussed below). The small model would hence begin training at a slower rate.

In the 5% test, the accuracy for the large linear model (U+L) increased to 82.23% and the medium linear model (L) achieved an accuracy of 82.02%, both of which represented significant improvements from the previous test (1% labeled data). The accuracies of both the large linear model (L) and the medium linear model (U+L) were also significantly increased at 76.08% and 78.09%, but the small model fell behind. The accuracies for the small model increased comparably as well but the absolute accuracies, at 69.79% (U + L) and 60.48% (L), were lesser than those of the medium and large models. As seen from Table 1, these accuracies for the small model would represent a small decrement from their maximum test accuracies. As shown in Table 2, the accuracies were larger when the percent labeled data increased from 0.01% to 5% when compared with when the percent labeled data increased from 1% to 5%. However, when these accuracies were normalized to the increase in the percent labeled data as a ratio, it can be seen that the increases in accuracy for a 5 fold increase in percent labeled data (1% to 5%) were an order of magnitude greater than the increases in accuracy for a 500 fold increase in percent labeled data (0.01% to 5%). Hence, accuracy increases, when expressed in terms of a unit increase in percent labeled ratio, were greatest when going from 1% to 5% labeled data.

Table 2. Increase in accuracy as a function of the ratio of the change in the percent labeled data

	Large model		Medium model		Small model	
	U+L	L	U+L	L	U+L	L
Increase in Percent labeled data as ratio	K-fold increase in accuracy					
5/.01=500	8.14	7.8	5.04	4.94	4.91	6.32
5/1=5	1.20	1.25	1.18	1.6	1.16	1.11
	Large model		Medium model		Small model	
	U+L	L	U+L	L	U+L	L
Increase in percent labeled data as ratio	K-fold increase in accuracy / increase in percent labeled data as ratio					
5/.01=500	0.016	0.016	0.010	0.010	0.010	0.013
5/1=5	0.24	0.25	0.24	0.32	0.23	0.22

Extrapolating this concept, we plotted a graph with the ratio of the labeled percent data at any point to its previous value on the X-axis versus the ratio of their corresponding accuracies on the Y-axis. The intersection of two best fitting lines for different data ranges could be then construed as the 'optimal' threshold of accuracy for that model. For example, Figure 1.4 represents the data for the (U+L) large model. It can be seen that the optimum threshold of accuracy occurs at ~2.7% of the percent labeled data for this model. We similarly obtained the optimum threshold of accuracy for other models. The results are presented in Table 3. It must be emphasized that although these numbers are approximations, they do represent two different regimes under whom the accuracy is an order of magnitude different (see Table 2). Furthermore, all the models yield approximate values of between 2.7% and 3%, above which the accuracies increase by an order of magnitude for all the models.

Table 3. Accuracy thresholds for different models

Model	Large (U+L)	Large (L)	Medium (U+L)	Medium (L)	Small (U+L)	Small (L)
Accuracy threshold (percent labeled data)	2.7	2.7	3.0	2.7	2.7	2.8

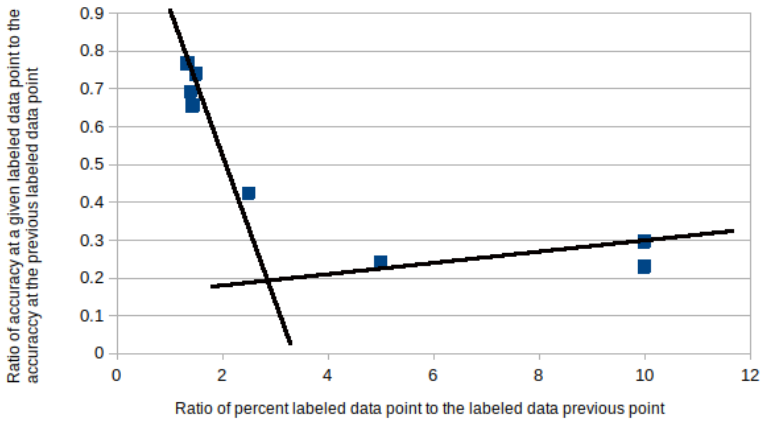

Figure 1.4. Using two regimes of accuracy increases to find the optimum percent labeled data above which acceptable accuracies may be obtained using that model. This figure shows the data from the (U+L) large model.

The next series of tests at 7% through 50% labeled data saw changes in the range of in the 74% to 92%, 72% to 85% and 56% to 76% for the large, medium and small models respectively. Going from 5% to 10% labeled data saw no changes in the accuracies, except

for the large L model, for which the accuracy increased significantly (although not by an order of magnitude when expressed per unit change in percent labeled data). This meant that the 5% data at or below 5% (but > 1%) labeled data could be viewed as an "optimal" threshold for accuracy at this level. However, the large linear model was able to reach almost 90% accuracy after the 15% and 20% tests. The medium and small linear models remained confined to 80% and 70% test accuracies respectively.

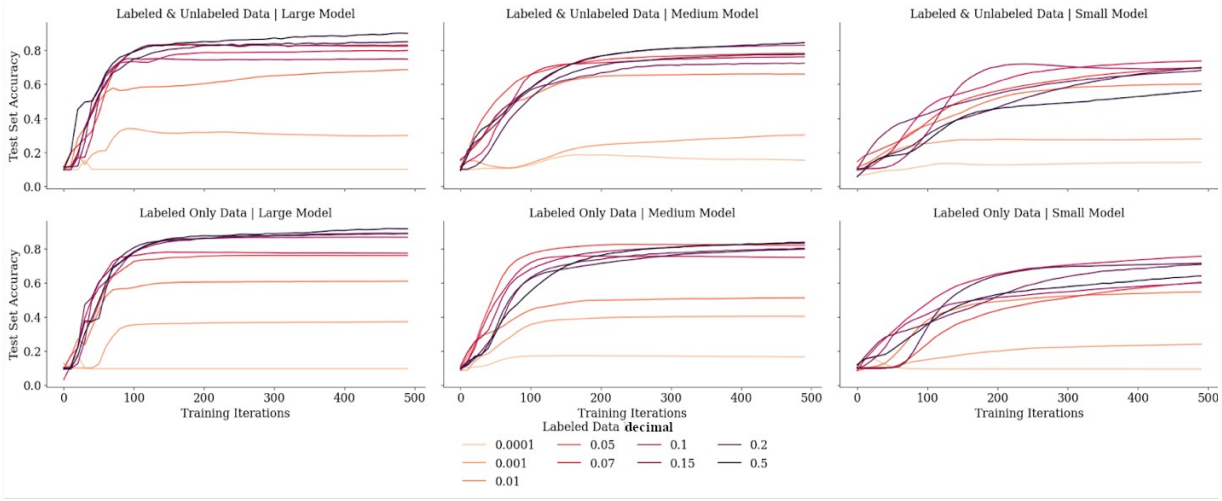

Figure 1.5. Plots of Labeled Data Percentage and Training Iterations for Model Size and Data Condition

The graphs above represent the learning patterns of each of the models, large, medium, and small sorted by L and U+L Data. By far, the large (L) linear model had the steepest and sharpest curves across the models, an indication of fast convergence rates and high accuracies. In contrast, the other models exhibited moderately steep convergence curves, suggesting a lower learning rate and obtainable accuracies. Of the L data tests, the models exhibited smoother curve patterns over the iterations than their U+L counterparts.

However, the U+L data tests presented with different trends. The models all obtained similar accuracies as their L data counterparts, but the patterns were not smooth. In fact, these tests showed fluctuations in test accuracies even at high training iterations. Though this trend was present in all of the models, the small linear model (U+L) compared to the small linear model (L) demonstrated these instabilities well. The unlabeled data caused the learning rates to deviate from a smooth pattern, and at times, show a decrease in accuracies with an increase in iterations. Though all the models were affected negatively by the introduction of the unlabeled data, the small linear model was especially affected by it.

Additionally, there was a unique and unexpected relationship between the model sizes and their labeled data percentages. At the very low-test percentages, (less than 0.1%) , the test accuracies were about the same in all the models. The performance hovered at ~ between 10% to 40% for all models, regardless of complexity. With a slight jump in labeled data percentages, the accuracies of the medium model track closely to those of the large model with little difference. The accuracies of the small model falls behind the other two models rather quickly and levels off sooner, but the medium and large models have similar structures. With a medium amount of labeled data (~ 10%), the medium model performs better or equal to the large model despite the large model's faster learning rate. When the labeled data percentage jumps again and uses a substantial percentage of labeled data, only then is the large model most accurate.

After the labeled data percentage increased 5%, the models did not make significant accuracy jumps. As seen in the medium model, from tests 5% and up, the test set accuracy crowded around 80%. The difference between 5% and 50% accuracies is not distinctive in each of the models considering the amount of increased labeled data. This created a threshold of accuracy at about 80% when labeled data percentages were between 5-10%.

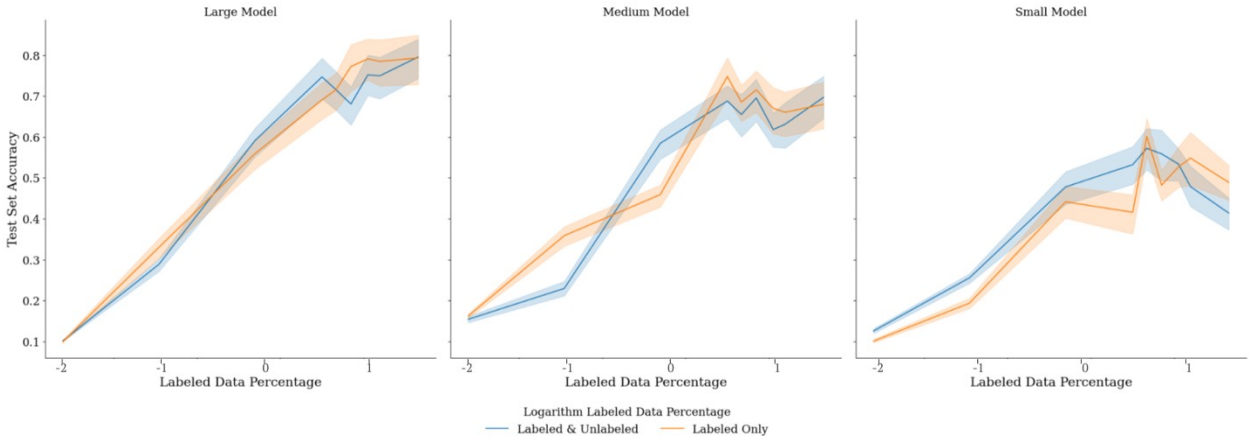

Figure 1.6. Plots of Test Accuracy and Log Labeled Data Percentages Per Model Size

Another interesting trend was discernible between the labeled data percentages and the use of only labeled (L) or labeled and unlabeled (U+L) data. Figure 1.6 depicts the test set accuracy versus the labeled data percentages, graphed using a moving average of the data table. At the start, for a limited amount of labeled data, the L data outperformed the U+L data, except for the small model, where the U+L data was more accurate than the L data but the difference was insignificant. For a limited amount of labeled data, the introduction of unlabeled data may have added noise that could hinder rather than assist the learning process. However, as the labeled data percentage increased, there was a

sufficient quantity of labeled data such that the introduction of unlabeled data became beneficial. This meant that the model could leverage the additional unlabeled data to achieve better accuracy. In fact, the U+L data only achieved a higher accuracy than the L data for a specific range of percentages of labeled data which created "sweet spot" for semi-supervised learning with linear models. From Figure 1.6, this range occurs ~ between -0.5 and +0.8 on the log scale; or between 0.32% and 6.3% percent labeled data. In this phase, the model benefited from using both labeled and unlabeled data to help it reach higher accuracy than labeled data alone. As the labeled data percentage increased, the L data achieved higher accuracies and the benefits of the unlabeled data diminished. Because unlabeled and labeled data began as lower than L data, rose above it, and then returned back below it, it created an "S" shaped learning curve---an optimum performance for U+L data versus L data. These "S"-shaped learning patterns were especially distinctive in the large and medium models above.

Discussion

The results of our experiments emphasized the effect of labeled data percentage, model complexity, and "S"-shaped learning patterns. One of the key results from our analysis was the threshold of unlabeled and labeled data. We found that if a small percentage of labeled data was used, models should be trained on solely labeled data. Unlabeled data should be utilized when the labeled data increases. The "sweet spot" for using unlabeled data was around 0.9-2% of labeled data. The introduction of unlabeled data in this regime boosted the model's performance when compared to using L data. However, past 3% the model should use L data again as the unlabeled data became less effective in test set accuracy. Understanding this threshold is important for making decisions on resource allocation. In scenarios with a large amount of labeled data, focusing solely on labeled data may be more efficient. Conversely, in data-scarce situations, leveraging unlabeled data can lead to significant accuracy improvements as seen in the "S"-shaped graphs in the Results section.

Next, we identified the relationship between SSL and model complexity. When labeled data was limited, the medium linear model outperformed the large linear model despite its faster learning rate. While complex models, like the large linear model, use labeled data effectively, it can be difficult for them to perform using small amounts of data. When data is significantly limited, such as in the 0.01% and 0.1% experiments, using a moderately complex model, like the medium model, may extract more meaningful information from the data. However, as labeled data percentages increase past 1%, the large linear model regains its superiority and its complexity can capitalize on the included labeled data, resulting in a higher accuracy.

Despite these successes, there are apparent limitations to the generalization of this research. Our results correspond to a neural network with five layers, two of them being ReLu and the others being linear. This may affect neural networks with more or less layers adversely, compared to our results. Other neural network architectures or networks with

different hyperparameters may respond differently than that of those we obtained in this research.

These findings allow practitioners to try to use less labeled data while continuing to achieve high model performances. While the "sweet spot" for each model may vary, the concept that unlabeled data can improve model performance in a specific regime of percent labeled data remains. Practitioners should also choose model complexity based on the amount of labeled data available to yield better results. In data-scarce scenarios, opting for a moderately complex model rather than a more complex model will benefit model accuracies. These insights allow practitioners to make further informed decisions on how to best leverage available data resources and design effective ML models for real-world applications.

Conclusion

In this study, we experimented with autoencoders and Semi-Supervised Learning to address the common difficulty of learning from high-dimensional image data. Our findings revealed that there was a 'sweet spot' for SSL with linear models in moderately labeled data regimes as seen in the "S"-shaped learning curves. When there was little to no labeled data, using only labeled data resulted in higher test accuracies. However, as labeled data increased, introducing unlabeled data accrued beneficial results. Lastly, model complexity played a large role in determining how labeled and unlabeled was utilized. In data-scare situations, medium-sized models performed with greater accuracy on small labeled data percentages but as this labeled data percentage increased, more complex models could better capitalize on the labeled data available.

There are several avenues for future research. Extending our experiments to use a more sophisticated autoencoder architecture, such as that of the variational autoencoder could enhance learning capabilities through its disentanglement potential (15). Using datasets with real-world deployments, such as healthcare or autonomous vehicle images, can help assess the generalizability of our research. On the other hand, a more in-depth exploration of the model hyperparameters, such as learning rates, batch sizes, optimization algorithms, and model depths, could further optimize model performance. Studying these parameters can help identify the best configurations for the various levels of supervision.

As ML expands into various domains, developing accurate models with little supervised data is increasingly important. The pursuit of interpretable, efficient, and ethical models remains a collective goal to advance the capabilities of machine learning in diverse applications. Our work in examining Semi-Supervised Model performance under various label settings offers a first step in understanding how to achieve these goals.

References

1. Chen, T., Chen, H. (1995). Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE transactions on neural networks*, *6*(4), 911-917. https://ieeexplore.ieee.org/document/392253

2. Jiao, L., Zhang, R., Liu, F., Yang, S., Hou, B., Li, L., et al. (2021). New generation deep learning for video object detection: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(8), 3195-3215. https://ieeexplore.ieee.org/document/9345705

3. Berisha, V., Krantsevich, C., Hahn, P. R., Hahn, S., Dasarathy, G., Turaga, P, et al (2021). Digital medicine and the curse of dimensionality. *NPJ digital medicine*, *4*(1), 153. https://www.nature.com/articles/s41746-021-00521-5

4. Yang, X, Song, Z, King, I, Xu, Z. "A survey on deep semi-supervised learning." *IEEE Transactions on Knowledge and Data Engineering* 2022. https://arxiv.org/abs/2103.00550

5. Y. Chen, Mancini, M, Zhu, X, Akata, Z."Semi-Supervised and Unsupervised Deep Visual Learning: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. https://arxiv.org/abs/2208.11296

6. Zhang, C, Bengio, S, Hardt, M, Recht, B, Vinyals, O. "Understanding Deep Learning Requires Rethinking Generalization," *International Conference on Learning Representations* 2017. https://arxiv.org/pdf/1611.03530.pdf

7. Bengio, Y, Courville, A, Vincent P. "Representation learning: A review and new perspectives." *IEEE Transactions on pattern analysis and machine intelligence* 35.8 (2013): 1798-1828. https://arxiv.org/abs/1206.5538

8. Sewak, Mohit, Sanjay K. Sahay, and Hemant Rathore. "An overview of deep learning architecture of deep neural networks and autoencoders." *Journal of Computational and Theoretical Nanoscience,* 2020. https://www.researchgate.net/publication/339480272_An_Overview_of_Deep_Learning_Architecture_of_Deep_Neural_Networks_and_Autoencoders

9. He, K, Chen, X, Xie, S, Li, Y, Dollar, P, Girshick, R. "Masked autoencoders are scalable vision learners." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022. https://arxiv.org/abs/2111.06377

10. Hendrycks, D, Mazeika, M, Kadavath, S, Song, D. "Using self-supervised learning can improve model robustness and uncertainty." *Advances in neural information processing systems* 32 (2019). https://arxiv.org/abs/1906.12340

11. Arazo, E., Ortego, D., Albert, P., O'Connor, N. E., McGuinness, K. (2020, July). Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*(pp. 1-8). IEEE. https://arxiv.org/abs/1908.02983

12. Sohn, K, Berthelot, D, Li, C, Zhang, Z, Carlini, N, Cubuk, E, et al."Fixmatch: Simplifying semi-supervised learning with consistency and confidence." *Advances in neural information processing systems* 33 (2020): 596-608. https://arxiv.org/abs/2001.07685

13. He, K, Fan, H, Wu, Y, Xie, S, Girshick, R. "Momentum contrast for unsupervised visual representation learning." *Proceedings of the IEEE/CVF Conf. on computer vision and pattern recognition*. 2020. https://arxiv.org/abs/1911.05722

14. Zhu, Xiaojin. "Semi-supervised learning literature survey." 2005. https://pages.cs.wisc.edu/~jerryzhu/pub/ssl_survey.pdf

15. Pu, Y., Gan, Z., Henao, R., Yuan, X., Li, C., Stevens, A., Carin, L. (2016). "Variational autoencoder for deep learning of images, labels and captions." *Advances in neural information processing systems*, *29*. https://arxiv.org/abs/1609.08976