Enhancing Classification Performance through Advanced Autoencoder Architectures and Discriminative Latent Representations

James Dooley
Department of Electrical & Computer Engineering
University of Florida
Gainesville, FL, USA
james.dooley@ufl.edu

Tre' R. Jeter
Department of Computer & Information Science & Engineering
University of Florida
Gainesville, FL, USA
t.jeter@ufl.edu

***Abstract* – This project explores advanced strategies to enhance classification performance using autoencoder architectures and discriminative latent representations. In the first phase, a stacked autoencoder (SAE) with multiple hidden layers is trained on the Kuzushiji-MNIST (K-MNIST) dataset. The SAE's bottleneck layer outputs serve as features for subsequent classification using a Multi-Layer Perceptron (MLP) classifier. The impact of varying the bottleneck layer size on performance is systematically investigated and compared against a baseline MLP classifier. Additionally, impulsive noise is introduced to the input images, and the SAE is trained with (1) Mean Squared Error (MSE) and (2) correntropy cost functions to mitigate the noise effect. The second phase involves designing a penalty function to enhance discrimination in the latent space by introducing a constellation-based regularizer. The constellation targets are defined, and the cost function integrates distance measures to guide the autoencoder training. The project systematically evaluates the impact of regularization strength on classification performance.**

***Index Terms*—Neural Networks, Multi-Layer Perceptron, Stacked Autoencoders.**

I. Introduction

In the ever-evolving landscape of machine learning, the significance of autoencoder architectures, particularly stacked autoencoders (SAEs), and Multi-Layer perceptrons (MLPs) cannot be overstated. These neural network structures have proven instrumental in real-world applications, showcasing their adaptability and efficiency in tasks ranging from image recognition to data compression. SAEs, in particular, excel in feature extraction and representation learning, providing a powerful framework for enhancing the discriminative capabilities of models. On the other hand, MLPs, with their deep learning capabilities, have demonstrated prowess in capturing complex patterns and relationships within data, making them formidable contenders in various applications.

However, as we navigate the intricacies of real-world scenarios, the need for advancing these models becomes apparent. Challenges such as classification accuracy, robustness to noise, and interpretability demand continuous refinement of machine learning architectures. This project fills a crucial gap by undertaking a comprehensive exploration of SAEs and MLPs, seeking to optimize their performance and adaptability. The unique focus on introducing impulsive noise, coupled with the novel approach of constellation-based regularization, reflects a commitment to addressing the nuanced demands of real-world data.

The expected results from the comparative analysis between SAEs and MLPs in our specific task carry profound implications for real-world development. If SAEs demonstrate superior discriminative capabilities and resilience to noise, it could reshape the landscape of applications requiring robust feature extraction and classification. On the other hand, if MLPs exhibit comparable or even superior performance, it reaffirms their standing as reliable workhorses in deep learning. Understanding how these models fare in our specific context offers valuable insights into their suitability for real-world applications, steering the trajectory of future developments in machine learning.

Note: Mr. Dooley handled Task 1 of the project while Mr. Jeter handled Task 2.

Organization. The rest of the report is organized as follows: Section II describes our methodology to carry out our experiments. Section III explains our experimental results. Section IV is an open discussion about expectations and our comparative analysis outcomes of SAEs and MLPs in regards to our task. We also discuss our minimal ablation study of the SAE in Task 1 and Task 2 when adding the penalty. Lastly, Section V concludes the report.

II. Methodology

A. Designing the Stacked Autoencoder For Use in an MLP

The approach to setting up the Stacked Autoencoder (SAE) and Multi-Layer Perceptron (MLP) involved a systematic methodology. Initially, a comprehensive exploration of hyperparameters was conducted for the SAE, testing varying bottleneck layer sizes and activation functions against different loss functions. Table I depicts the bottleneck layer sizes, activation functions, and loss functions tested. Subsequently, a series of iterative experiments were performed to determine the optimal encoding layer size for integration into an MLP. This involved exploring a range of encoding dimensions. Notably, the best performance was identified with an encoding dimension of '81', prompting two additional rounds of testing to further refine the optimal layer (49 and 64). The exploration extended to MLP-related features, including varying MLP sizes , the number of hidden layers, and activation functions (ReLU, Tanh, SeLU). Table II depicts each hyperparameter tested for the optimal MLP design. The process concluded with the identification of the best-performing network configuration,

with an illustrative example being an encoding dimension of 64, MLP size of 4096, and a single layer using the SeLU activation function. Figure 1 shows the relationship between loss and encoding dimensions with various activation functions. Additionally, a denoising step was incorporated using Mean Squared Error (MSE) and correntropy, with the optimal sigma parameter visually determined. Both networks' encoding outputs were utilized in training the final MLP.

Bottleneck Layer Sizes	Activations	Loss
2-784	Tanh, ReLU, SeLU	MSE, Correntropy

TABLE I: Hyperparameters Tested for Optimal SAE Design. *Note*: Bottleneck layer sizes were increased as squares (i.e., 2, 4, 8, 16, 32...). The encoding layers of the SAE are (800×200×XXX) where XXX represents the bottleneck layer.

Encoding Dims.	Size	No. Hidden Layers	Activations
49-324	512-8192	1 or 2	Tanh, ReLU, SeLU

TABLE II: Hyperparameters Tested for Optimal MLP Design. *Note*: Encoding dimensions were all squared values (i.e., 49, 64, 81, 100...). MLP sizes vary similarly (i.e., 512, 1024, 2048...).

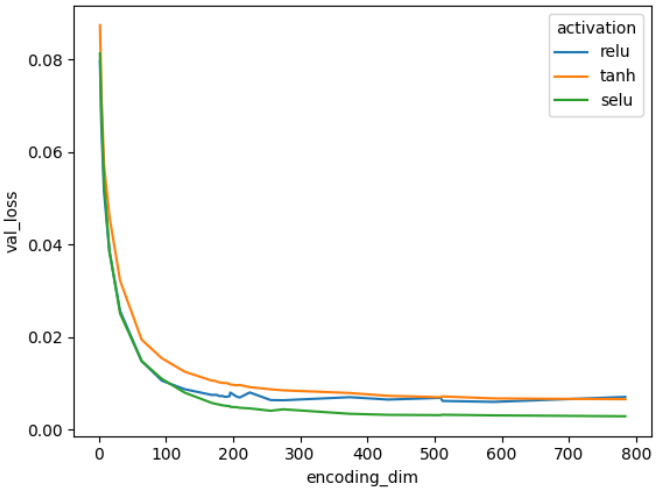


Figure 1: Visual Effects of Activation Functions on Loss with Increasing Encoding Dimension Size.

B. Designing the Penalty Function

We design a constellation-based regularization function that uses distances to enhance discrimination in the latent space. Because this is a classification problem, we know the number of classes (K-MNIST = 10 classes). Knowing the number of classes allowed us to create a set of constellations (priors) based on each class. Seeing that the most optimal bottleneck layer was size 64, we inherit this encoding dimension for our experiment. For each class, we calculate the average encoding and use these averages as our priors in our cost function. In doing so, we are able to calculate the average distance between the encoding of each class and add a regularizer to further enhance discrimination in the latent space. With this, we compute the reconstruction loss (MSE) and the regularization loss based on the latent space. Adding both losses gives us the total loss of the system. Mathematically, our loss function calculation is as follows:

MSE:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}(y_{\text{pred}_i} - y_{\text{true}_i})^2 \tag{1}$$

where N is the number of samples and i is the class.

Regularization on Constellations:

$$\mathcal{R} = \lambda\frac{1}{N}\sum_{i=1}^{N}(y_{\text{pred}_i} - enc_{\text{avg}_i})^2 \tag{2}$$

where λ is the regularization parameter and enc_{avg} is the average encoding vector for a class.

Total Loss:

$$\mathcal{L}_{\text{total}} = \frac{1}{|\lambda_v|}\sum_{\lambda\in\lambda_v}\left(\frac{1}{N}\sum_{i=1}^{N}(y_{\text{pred}_i} - y_{\text{true}_i})^2 + \lambda\frac{1}{N}\sum_{i=1}^{N}(y_{\text{pred}_i} - \text{enc}_{\text{avg}_i})^2\right) \tag{3}$$

where λ_v are the different values of λ tested. More simply put,

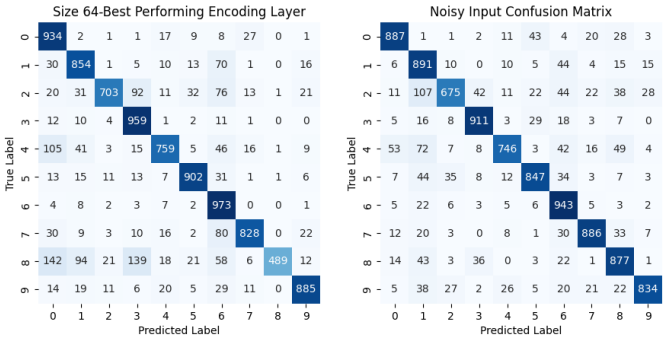
$$\mathcal{L}_{\text{total}} = \frac{1}{|\lambda_v|}\sum_{\lambda\in\lambda_v}(\mathcal{L} + \mathcal{R}) \tag{4}$$

III. Experimental Results

A. Best Parameters to Train SAE

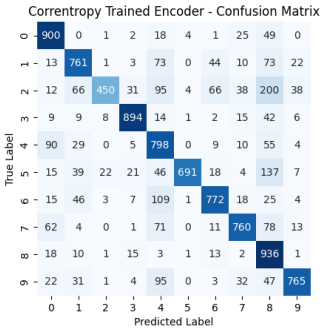
For the optimal encoding dimension experiment, we achieved remarkable results, with training accuracy reaching 98.61% and validation accuracy at 92.4%. Figure 2a depicts the confusion matrix with 64 as our most optimal encoding dimension size. When considering the noisy input, the model demonstrated robust performance, achieving a training accuracy of 98.6% and a validation accuracy of 91.6%. Figure 2b depicts the models confusion matrix after adding noisy inputs. Further refinement through correntropy-based training of the encoder yielded a slightly reduced, yet still impressive, training accuracy of 98.3% and a validation accuracy of 91.85%. Figure 2c depicts the models confusion matrix after implementing the model with correntropy.

In the exploration to determine the best sigma value, visual representations are provided in Figure 3. The examples include the original input, the corresponding noisy input, and instances at sigma values of 0.6, 0.75, 0.8, 0.9, 1, 1.2, 1.5. The sequence concludes with an example incorporating Mean Squared Error (MSE). This comprehensive exploration helps illustrate the impact of different sigma values on the denoising process and can be seen in Figure 4.



(a) Confusion Matrix with Best Encoding Size.

(b) Confusion Matrix with Noisy Inputs.



(c) Confusion Matrix with Correntropy.

Figure 2: Confusion Matrices for Various SAE Design Parameters.

Figure 3: Visual Effects of Varying Values of Sigma. *Note*: First two images are the original and noisy input.

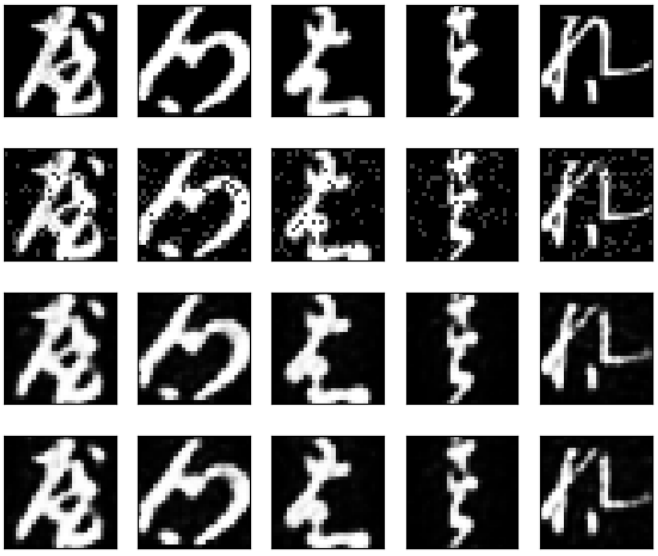


Figure 4: Effects of Sigma w/ Correntropy vs. MSE. **Rows from Top to Bottom**: Original, Noisy Inputs, Correntropy, MSE.

B. Experimenting with Different Regularization Parameters

We experimented with a wide range of regularization parameters to watch the effect the penalty has on the loss and accuracy. The test values of λ are presented in Table III. It is mentioned earlier that an encoding dimension of 64 resulted in optimal results for the SAE. When classifying our data, the MLP classifier size that yielded the best results was 4096, therefore when presenting the classification results, we adopt this classifier size here as well. For simplicity and space constraints, we present our top two results When λ = 0.002 in Figures 5-7 and 0.001 in Figures 8-10, respectively. Table IV reports large λ's performance and our two best cases to show the effects on accuracy related to λ. We see that our penalty function allows our model to reach a higher training and validation accuracy than the original model. This infers that our added penalty is executed as intended.

Values of λ
$[0.5, 0.1, 0.05, 0.01, 0.001, 0.002, 0.003, 0.004, 0.005, 0.0001]$

TABLE III: Tested regularization parameters.

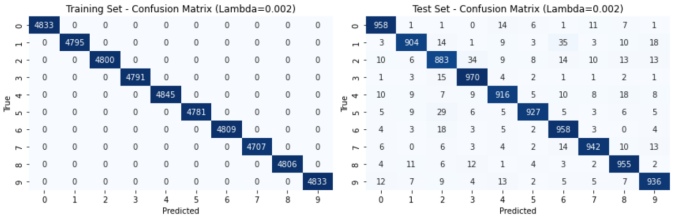


Figure 5: Confusion Matrices w/ λ = 0.002

λ	0.5	0.1	0.001	0.002
Training	83.09	89.23	100.00	100.00
Validation	70.57	77.25	93.44	93.50

TABLE IV: Effects of λ on Performance

IV. Discussion

Expectations. The expectation that the SAE+Classifier model would be outperformed by an identical model incorporating

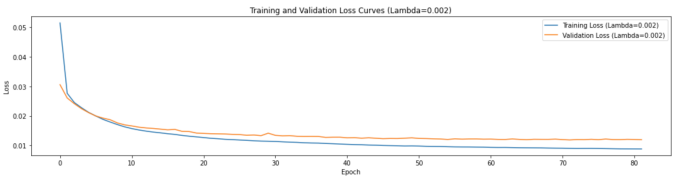


Figure 6: Loss Curves w/ λ = 0.002

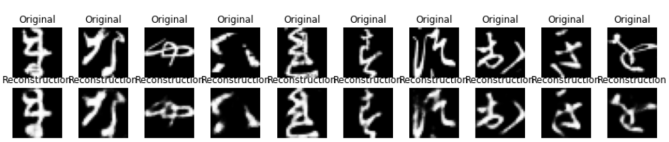


Figure 7: Reconstructions w/ λ = 0.002

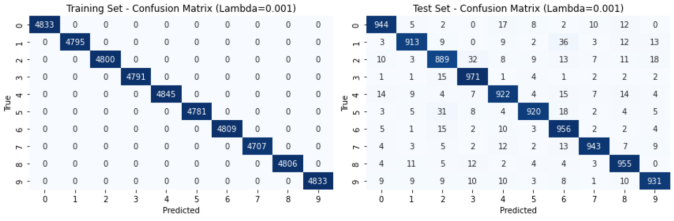


Figure 8: Confusion Matrices w/ λ = 0.001

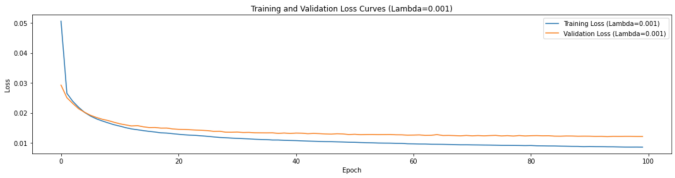


Figure 9: Loss Curves w/ λ = 0.001

regularization and utilizing a cost function based on average distances in the encoding space is well-founded. Regularization techniques are integral for preventing overfitting and promoting generalization in ML models. In the context of autoencoders and classifiers, regularization serves to encourage the learning of robust and adaptable representations. When the regularization is specifically tailored to penalize overly complex models, it guides the model towards a more generalized solution. In regards to the optimal size for the encoding layer for use in classification, it was originally supposed that the optimal size would be around 200 neurons, as that was the layers that had the best size-to-reconstruction-loss ratio, meaning it was the smallest network capable of completely reconstructing the original image with nearly no loss. This expectation was of course upended when smaller encoding layer sizes proved more effective as the input to an MLP.

Furthermore, the use of a cost function that accounts for the average distances in the encoding space for each class acts as a form of regularization itself. This approach encourages the model to encode class-specific information in a manner that minimizes overlap between different classes. By doing so, the model is guided to learn distinct and well-separated representations for each class. Consequently, the incorporation of both regularization and a class-specific cost function is expected to result in a model that is not only more discriminative but also more robust and adept at handling unseen data. Therefore, the expectation that the enhanced model would outperform its identical counterpart is grounded in the principles of regularization and effective encoding space utilization.

Outcomes. The outcomes of our experiments largely align with our expectations, providing validation for the effectiveness of the added penalty in our model. The model augmented with the penalty term not only achieved a perfect training accuracy of 100%, indicating a strong capability to capture the training data, but also demonstrated superior performance on the validation set with an accuracy of 93.50%. In comparison, the original model, which lacked the regularization mechanism, attained a slightly lower training accuracy of 98.6% and a validation accuracy of 92.4%. These results affirm that the incorporation of the penalty term, designed to enhance discrimination in the latent space, indeed contributes to improved generalization and classification accuracy, as anticipated.

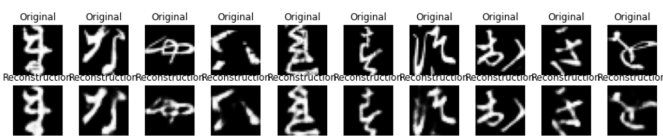


Figure 10: Reconstructions w/ λ = 0.001

Comparison. Of the three models trained in Part 1, the first model trained with MSE and noiseless inputs had the best performance on the test set. More interesting however, is that while performance between the three networks is quite comparable, the confusion matrices show that each network had differing abilities to distinguish certain classes. For instance the first model has great difficulty identifying class 8, often confusing it for class 0, while the model trained with correntropy frequently confuses class 2 for class 8. Meaning these classes likely sit close together in the encoding space, and the model is having a difficult time separating the classes consistently. When comparing correntropy and MSE in regards to their effect on a networks ability to reduce the noise present in a signal, our experimentation showed equivalent performance between the two error functions, and as seen in Figure 4, with both networks proving very capable of cleaning noise from an input signal, with indistinguishable results. Compared to our MLP from Project 1, our SAE+Classifier method far outperforms it. The optimal MLP from Project 1 achieved 95.48% and 92.91% on training and validation, respectively. Our MLP enhanced with an SAE here demonstrates a 3.13% increase in training performance. Moreover, with the penalty added, our model achieves even better results, boosting training and validation performance by 4.52% and 0.59%, respectively.

V. Conclusion

In conclusion, our project successfully identified optimal configurations for a Stacked Autoencoder (SAE) combined with a Classifier (MLP) and applied it to the K-MNIST dataset. The addition of a regularization penalty, determined by a cost function relying on the average distances within the encoding space for each class, yielded notable improvements over the original model. The enhanced model achieved superior training and validation accuracies, demonstrating its ability to better generalize to unseen data. This finding suggests that the regularization mechanism, designed to encourage distinct and discriminative encoding representations, positively impacts the model's overall performance.

In real-world applications, the observed improvements have significant implications. The model's increased accuracy and generalization capabilities indicate its potential for more robust and reliable performance when applied to diverse datasets or real-world scenarios. This can be particularly crucial in tasks where accurate and nuanced classification is paramount, such as image recognition, medical diagnosis, or various other

machine learning applications. Overall, our project provides valuable insights into optimizing autoencoder-based classifiers and underscores the potential benefits of incorporating regularization techniques for improved real-world applicability.

VI. References

[1] Liu, W., Pokharel, P. P., & Principe, J. C. (2006, July). Correntropy: A localized similarity measure. In The 2006 IEEE international joint conference on neural network proceedings (pp. 4919-4924). IEEE. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=16dc2df0e7da25ab01c47816a7a0804323bfa20b

[2] https://stackoverflow.com/questions/70991353/custom-loss-function-in-keras-correntropy-math-implementation-issues

[3] https://keras.io/examples/generative/vq_vae/

[4] https://cnvrg.io/keras-custom-loss-functions/

[5] https://www.kaggle.com/code/peremartramanonellas/guide-multiple-outputs-with-keras-functional-api

APPENDIX

We broke this project up into tasks. These tasks can be viewed in Figure 11. We believe our experiments validate the completion of each task.

Tasks	Percent Complete
Build first SAE	100%
Build First SAE with tied weights	100%
Define Training Loop	100%
Gather images for encoder/decoder working	100%
Gather data for best layer size	100%
Generate Noisy data for training	100%
Build a Classifier off encoding layer data	100%
Build multiple classifiers with multiple encoding layer sizes/compare performance	100%
Compare MSE vs Correntropy for image cleaning	100%
Quantify the effect of added noise on the codes themselves, by comparing classification accuracy with the previous SAE+ Classifier and the versions trained with MSE and with correntropy	100%
Design a penalty function/regularizer on the encoding space using a prior	100%
Evaluate classification accuracy vs all other models with MLP	100%
validate regularization penalty with an experiment?	100%

Figure 11: List of Tasks.