Visualizing Deep learning Models for Natural Language Processing tasks: INFO VIZ 5602 Project Report

Student: Sushma Akoju*
University of Colorado Boulder

Professor: Prof. Abram Handler†
University of Colorado Boulder

ABSTRACT

It is well known that Image processing data is interpretable and explainable visually from Neural Networks. With a little more analysis, it is possible to explain Image data. This makes for a good reason why there are such vast number of visualization tools that visualize Image processing data over Neural networks or Deep Learning models. But when we have to visualize and explain Natural Language Processing (NLP) tasks, it gets challenging to visualize and explain. Can we visualize the learning process over epochs or layers in a network to see what it means to predict a word or predict a sentiment? What does it really mean to say this Deep learning model learnt to predict the sentiment of this sentence? In this project, I present analysis of research work from University of Arizona on Image processing data visualization over Convolutional neural networks (CNNs) and how the problem of visualization was solved. I then present an analysis of available visualization tools for NLP tasks and what visually explainable features are most desired for text, in comparison to Image processing tasks.

1 INTRODUCTION

It is often the case that when a new deep learning model is released, it gets difficult to understand and interpret the layers in the network for a naive or non-expert of Deep Learning models. Although Interpretability might be relatively complex for Deep Learning networks, to visualize weights from intermediate layers should not be a problem. The input data is numeric or Tensor or vector data which usually have very high dimensional Input data. Adding to such high dimensionality of the input data and the weights, is the type of data the input represents: textual, multivariate or image data types. To an extent, we could maybe intuitively visualize Image data transformations through each of the layers in the Deep Learning network. For example, we can show how image classification is done visually using a visualization package or research work such as [7]. However it gets challenging to see how textual data is learnt at each epoch, over each layer and how change in weights really affects the classification or prediction task. For example, one of the popular BERT Language Models, trained over a large text corpus, uses masks to replace words randomly and learn to predict the missing words from the contextual information. In an attempt to achieve that, predictions of Natural Language processing (NLP). Tasks can vary vastly by varying gender words such as replacing "he" with "she". In the gender bias case, suppose the sentence BERT is learning about is "$< Name >$ is angry". If we observe that BERT assigns higher or lower scores to this masked sentence when $< Name >$ is replaced with female names than that of male names, then this is called gender bias in the BERT model. Such a bias can be serious in nature if the corpus is a patientś Electronic Health record and that could impact the quality of resulting diagnoses. Thus it becomes important to assess and find explainability, interpretability in general for deep learning models.

Consider an example: "She is watching a movie on her laptop, while having Boba tea". Let us say, the random mask picked by BERT would be to replace Boba tea. Then the input sentence becomes: "She is watching a movie on her laptop, while having [CLS]". We would like to find out how the model is learning after a certain number of epochs versus specifically from BERT's transformer layer. BERT is used for several downstream tasks classification, language inference, Question & Answer systems and so on. The analysis for each task is a key step towards understanding why the model classifies or predicts differently. While this is an open research question, this makes for a good reason why this can be challenging to engineers, analysts who are not necessarily experts in deep learning models for analyzing language corpus. [9]

As discussed in [9], authors broadly categorized users into expert users and lay users. Expert users are the researchers who work on building Deep Neural Networks (DNNs) and lay users do not have knowledge about DNNs but build applications from DNNs by treating DNNs as blackbox.

Combining all of the above, i.e. general complexity of deep learning models, which serve as black box as well as additional complexity for understanding textual data transformations across various epochs/layers in a language model, makes this an interesting as well as challenging task. Thus in an exhaustive search to understand the complex nature of visual explanation in Deep Learning models for Natural language processing, I found several existing works and visualization tools. However there is no one-single-solution that visually explains such models. I found following visualization tools that attempt to explain deep learning models:

1. Tensorflow Playground
2. ANN Visualizer
3. Simple Neural Network visualizer VisualNN
4. Visualizing weights
5. TensorSpace
6. Intriguing properties of Neural Networks : A list of representations using visualization tools for Neural Networks [10]
7. Benchmark for Open Information Extraction [4]

1.1 Explainability

More recently, explainability has taken multiple directions:

1. Explaining something statistically and mathematically that supports hypothesis over Deep learning model activations, epochs and layers.
2. Explaining something visually to a data scientist or someone who has no idea about Deep learning as well as the Language processing tasks.

*e-mail: sushma.akoju@colorado.edu
†e-mail: abram.handler@colorado.edu

3. Trying to explain everything is going to be challenging anyway, just limit explainability to intersect between mathematical as well as visual explanations.

4. Use other more advanced approaches such as higher order logic (is research intensive) and is one direction I am in favor of. An example of this direction is [2] where the authors propose that a network can learn to explain as it learns to classify by making use of First Order Logic.

More specifically [9] suggest explanation methods into 3 categories:

1. Rule-extraction methods (deals with extracting rules that approximate the decision making process)

2. Attribution methods (deals with measuring importance of a component by changing the input or internal components)

3. Intrinsic methods (deals with interpretability, to understand and interpret internal structure of representations)

Rule extraction as per [9] can broadly be acheived by 3 subcategories:

1. Decompositional approach (deals with breaking down the network into individual parts)

2. Pedagogical approach (deals with viewing rule extraction itself as a learning task where target concept is the function computed by the network and input features are the network's input)

3. Eclectic approach (deals with "..membership in this category is assigned to techniques which utilize knowledge about the internal architecture and/or weight vectors to complement a symbolic learning algorithm")

1.2 Why is visualizing neural networks and Deep learning Neural networks difficult?

1. Lack of Contextualization: Generally in the first layer of neural network, we have an idea about input and weights, which are easier to visualize. However at hidden layers, it is difficult to find out what the network really is learning. [11]

2. Indirect Interaction: Some neurons from other layers/intermediate layers interact with each other. In other cases, neurons interact with each other through intermediate neurons. [11]

3. Dimensionality and Scale: There are so many neurons, hence many dimensions and scale. And it gets challenging to display to visualize a human scale amount of data. [11]

4. Deep learning models are complex and are referred to as "blackbox" due to the challenge in explaining textual data transformations within each network layer. What does it really mean to have textual data transformation? [11]

5. Lastly, visual modality is best to first tell anyone about what the data is and how a Deep learning model learns over training through layers. Secondly, visual modality also helps to identify certain problems in data, model and training process much more instantly, saving time and costs for wrong predictions, errors in network, and learning from errors.

1.3 Goals

Neural networks are always considered as less interpretable. One way to deal with existing state-of-the-art neural networks is to decipher the network layer by layer, step by step and understand the network. BERT (Bidirectional Encoder Representations from Transformers) is one of the most widely used, popular Language models ever since it was introduced in 2018, by Google. Many Natural Language Processing tasks that use BERT either need to alter or generate large data to train and get accurate predictions. Some key concepts that contribute to such interesting accuracies for such downstream tasks are intriguing and were discussed widely and explained. However, when BERT does indeed go wrong or brings out an underlying bias, can BERT be better trained and examined to find the root cause of the problem? If so, how do we do it? To understand and answer these questions, I think visualization is a great mode of communication. I thought this could be converted into three sub-tasks: 1) attempt to find the best way to visualize unique layers in the neural network 2) explain input/output of each layer with an example and 3) show how the dimensions of word vectors are transformed after attention layers. Besides the attempt to self-learning and understanding, I am more interested in learning the network to actually attempt to explain about Neural network so other users of a Neural network model, who are non-domain-experts in both NLP as well as Neural networks, can understand and debug why a task went wrong or better improve the model for a specific task.

1.4 What are the available Visualization tools ?

Tools explored and analyzed as part of this project:

1. Tensorflow Playground

2. ANN Visualizer

3. Simple Neural Network visualizer VisualNN

4. Visualizing weights

5. TensorSpace

6. Intriguing properties of Neural Networks : A list of representations using visualization tools for Neural Networks [10]

7. Benchmark for Open Information Extraction [4]

8. Text visualization Browser https://textvis.lnu.se/

1.5 What features can we expect to explain visually?

Given language corpus for Deep learning model, following are the features, would help any non-technical person to understand at same level as expert: Visualize Activations (what network saw) Visualize Weights (how network saw) Attributions (why the neuron fired) Predictions from layer Predictions from epochs Fired vs not fired neurons

1.6 Are there any visualization tools to visualize text corpus over Complex Deep learning models?

There maybe some tools, with limited features to visually explain the data such as following:

1. Tensorflow Playground

2. ANN Visualizer

3. VisualNN

4. Tensorboard

5. TensorSpace

6. quiver_engine

2 Visual Explanation for Image data over Neural networks

I replicated the research work [7], `https://github.com/distillpub/post--grand-tour` and I generated images from this replication of research work. This repository hosts same article with options to go over the data and model weights which were included as bin files.

In the process of understanding and exploring Neural Networks, I found this visualization for Image classification task by training Neural Networks. [7] and `https://distill.pub/2020/grand-tour/` was an interesting way of visualizing and telling a story of how model i.e. Network receives input image data (which is in vector of numbers, far from images human see), but perceives similar to that of humans (by using neural network layers) and eventually understands the features in the Image data (learning by feedforward, backtracking, learning from loss and adjusting weights). [7] To explain this visualization further, I would like to describe what and how this is described in this visualization process, attempt from University of Arizona team: First they use how systematically the digit classification improves from training. For this first the high-dimensional point clouds are projected to represent 10 dimensions (i.e. each dimension representing each of 10 digits in MNIST) in a two dimensional plot. They use Grand Tour, which uses a linear representation of multi-dimensional data unlike t-SNE or UMAP which are more popular for representing multidimensional data. Three use cases represented are :

1. visualizing the training process as the network weights change
2. visualizing the layer-to-layer behavior as the data goes through the network
3. visualizing both how adversarial examples

What is visualized in this research work? To explain this, the authors of the article selected a large dataset and neural networks from ImageNet Large Scale Visual Recognition Challenge. The authors believe that showing neural network activation and its influence from input to output transformation might be difficult to represent. Instead they attempt to show the changes in data between epochs,how classification of data converges (or diverge) when presented with different types of data.

Which Datasets were used? The three datasets trained on the same Neural Network are: MNIST, fashion-MNIST and CIFAR-10. A simple linear layers of classic Convolution Neural Network is displayed here over 100 epochs:

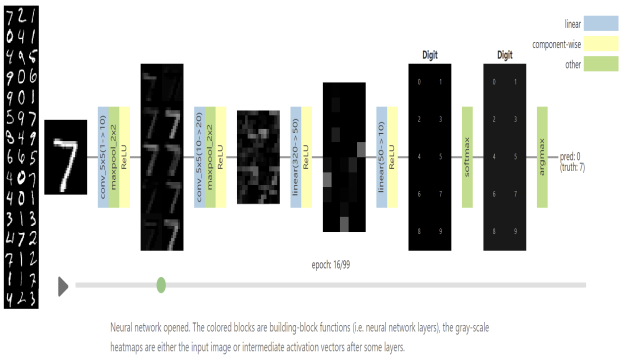

Figure 1: A visualization of Convolutional Neural network. [7]

The article generally follows Neural networks i.e. both convolutional and fully-connected networks with max-pooling, and ReLU 6 layers, culminating in a softmax 7 layer. The layers are broadly classified into linear transformations and simple nonlinear functions.

The above neural network example is provided to demonstrate how it serves as a bad example for interpreting the network layers and interactions.

The article then discussed how to visualize the training of deep neural networks. First article considers the loss, which is scalar value and how it is optimized and expected to decrease over epochs by using a gradient descent approach. The article plots flat training loss and then compares it with class-wise training loss plot. The per-class training loss plot, which is as follows, depicts that between epochs 14 and 21 the curve goes flat and then decreases. Specifically the flattened trend and then decreasing trend occurs for classes 1 and 7. Thus article suggests that on the whole we generally know neural network learn to recognize all the digits, but this class-wise learning trend shows that learning differs between numbers. It is not until epoch 14 that the network learns to recognize digit 1 and not until epoch 21, the network learns to recognize digit 7. More specifically the following plot suggests the class-wise error rate for learning each of digits.

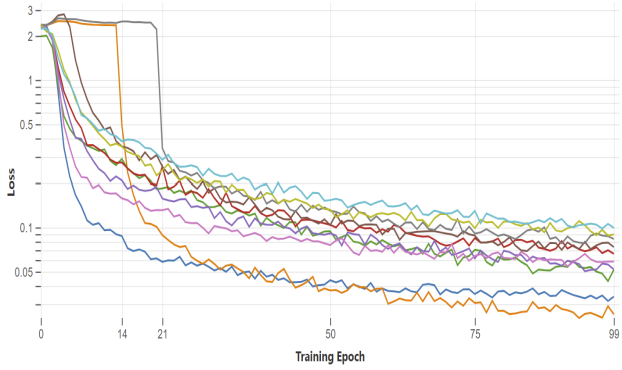

Figure 2: A visualization of Training epoch vs loss for each Digit class. [7]

The article follows that from above plot discussion, it is clear that activations of neurons (that is the last softmax layer) can provide significant insight into class-wise behaviour in learning to recognize digits and patterns of learning that impacts learning, optimizing loss over the epochs. Thus to visualize the softmax layer, it turns into a dimensionality reduction problem since the last layer here in the digit recognition task has 10 neurons. This can be an even larger number for different types of datasets and number of classes.

The article discusses t-SNE, UMAP, dynamic t-SNE and discusses the following plot, for representing dimensionality reduced data in 2-dimensional space. The article discusses how for digits in question i.e. digits 1 and 7 are represented for class-wise learning, loss behavior is visualized by the 3 types of visualizations used in this plot.

Digit 1: how the digit 1 is represented in epoch 13 and 14 by using each of the t-SNE, dynamic t-SNE and finally UMAP. By observing the behavior between each epoch for just one class, i.e. digit 1, it seems to show in UMAP representation that digit 1 which was clustered in epoch 13, suddenly turns into a well recognized single line in epoch 14.

Similarly for Digit 7 has epochs 20 and 21 which show the firing of neurons.

The article discusses the problem of visualizing non-linear

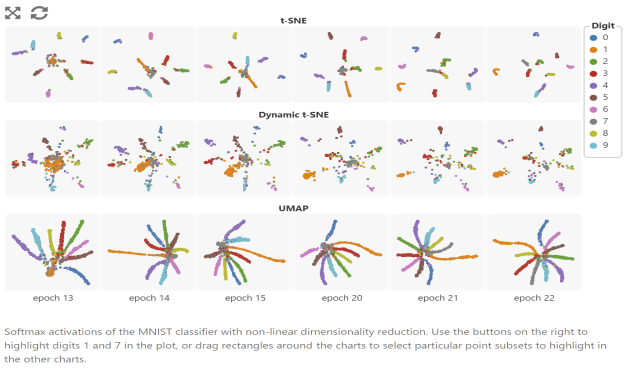

Figure 3: Class-wise Softmax activations for selected epochs. [7]

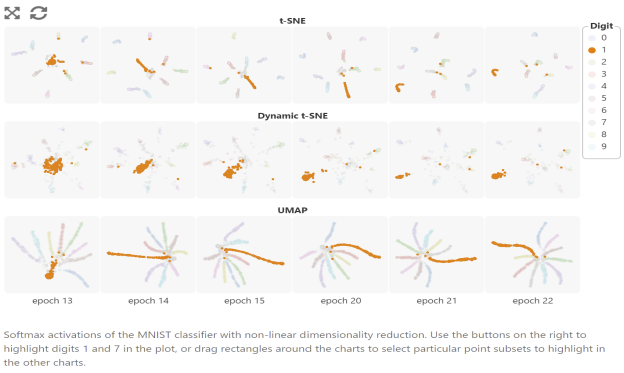

Figure 4: Digit 1 : Class-wise Softmax activations for selected epochs. [7]

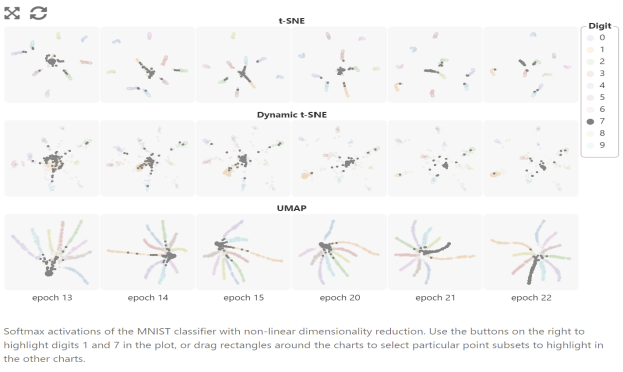

Figure 5: Digit 7: Class-wise Softmax activations for selected epochs. [7]

embeddings i.e. failure to represent and show such changes more clearly is due to data-visual correspondence principle. This principle as quoted in the article "...the principle states that specific visualization tasks should be modeled as functions that change the data; the visualization sends this change from data to visuals, and we can study the extent to which the visualization changes are easily perceptible". The change in data and visualization required to match the magnitude of change. t-SNE and UMAP do not adhere to data-visual correspondence principle, since the position of each single point depends on the entire data distribution from the embedding algorithm. From the following visualization between epochs 30 and 32, the article suggests how despite the stable neural network that already learnt is mis-represented by t-SNE, UMAP. Figures 6 and 61.

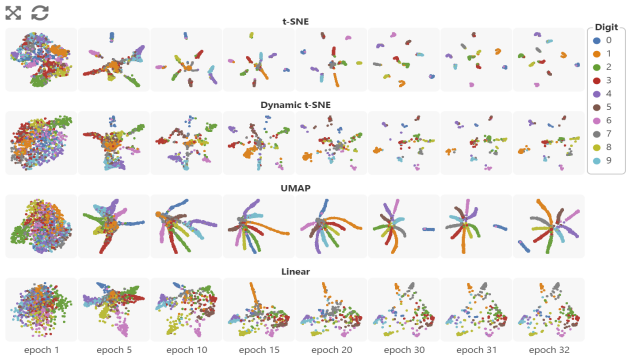

Figure 6: data-visual correspondence and violation with t-SNE, Dynamic t-SNE, UMAP and Linear (a method suggested in this research work): changes in data and visualization to match in magnitude. [7]

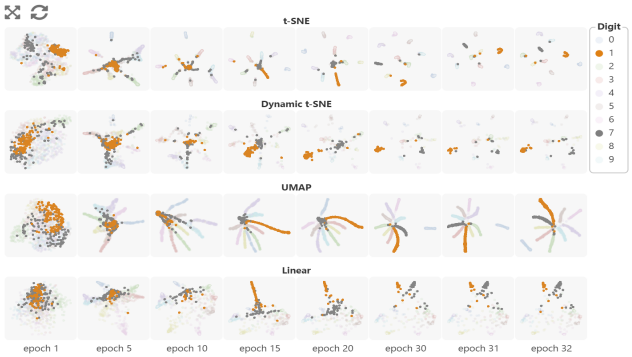

Figure 7: Digits 1 and 7: data-visual correspondence and violation with t-SNE, Dynamic t-SNE, UMAP and Linear. [7]

For another complex case, i.e. larger dataset such as Fashion MNIST, the article discusses how confusion among sandals, sneakers and ankle boots is represented by each of the networks. The article suggests that t-SNE and UMAP still visualize poorly about classification confusion. Figure 9.

Thus the article suggests that embedding algorithms such as UMAP, t-SNE for non-linear data could fail to show such confusion between classification for 3 classes as in the case of Fashion MNIST as follows. Thus the article concludes that linear projections provide a better way to represent the n-dimensional data, by actually converting each of n-dimensions to n 2-dimensional vectors. The example for digit 1 is discussed using The Grand Tour visualization. Digit 1 at epoch 13 and 14 as follows.

Digit 1 at epoch 14

A 10-dimensional vector is split into 10 2-dimensional vectors to show each vector's own behavior over 100 epochs. The above image is from epoch 13 and 14, which shows the behavior of the network in learning digit 1 and how it changes between epochs 13 and 14 and suddenly digit 1 is correctly classified. This is however an expected behavior due to activation functions such as softmax.

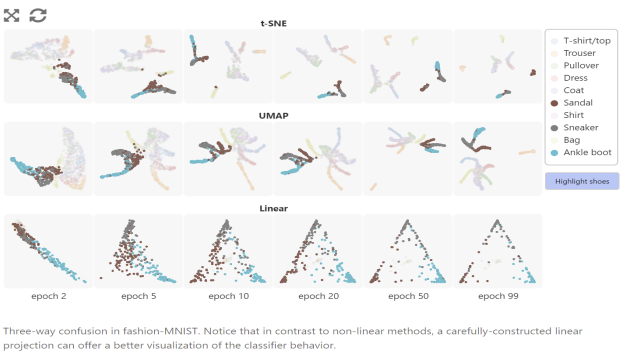

Figure 8: Sandals, Sneakers and ankle boots: data-visual correspondence and violation with t-SNE, Dynamic t-SNE, UMAP and Linear. [7]

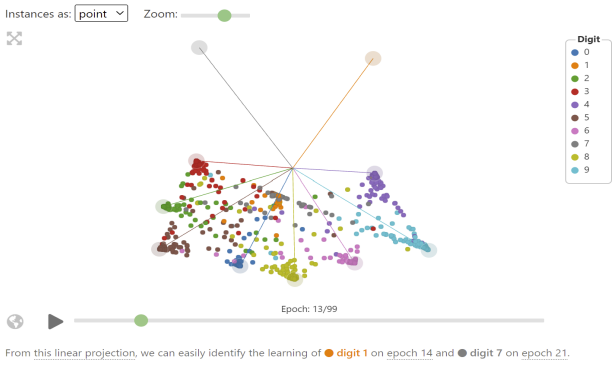

Figure 9: Linear projections for learning Digit 1 at epoch 13. [7]

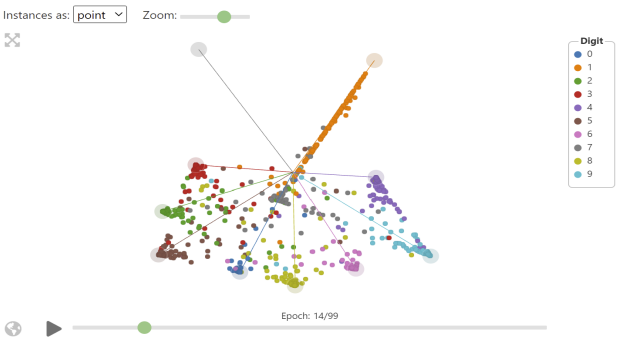

Figure 10: Linear projections for learning Digit 1 at epoch 14.

Similarly the same visualization using Grand Tour is used to represent and show 3 class confusion from Fashion MNIST. The confusion between sandals, sneakers and ankle boots is shown here. As networks learn 3 classes, the confusion is specifically between sneakers and sandals, then between sneakers and ankle boots. This however does not happen between all 3 at a time. The confusion is more specifically between 2 classes at any time.

The article finally proposes Grand Tour, which is a technique better than other dimensionality reduction methods such as Principal Component Analysis(PCA). PCA is good and reduces dimensionality however for the softmax layer already holds

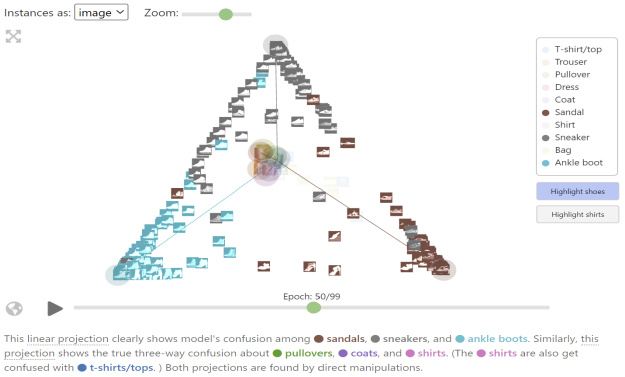

Figure 11: Epoch 50: 3-way confusion between sandals, sneakers and ankle boots. [7]

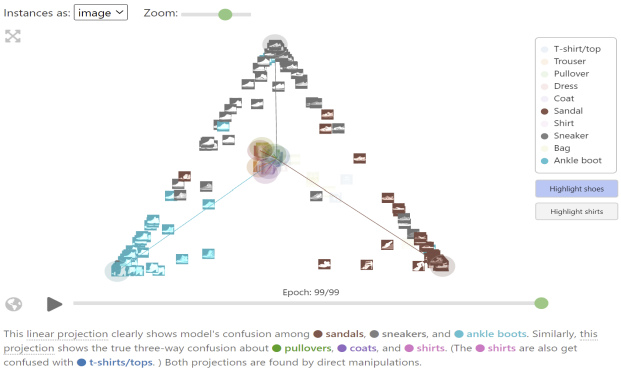

Figure 12: Epoch 99: 3-way confusion between sandals, sneakers and ankle boots. [7]

axis-specific semantics i.e. the softmax layer already learns information along each dimension i.e. for each class in classification. Attempting to reduce dimensions of such semantic rich dimensional data can adversely affect visualizing the data. Thus they suggest without loss of generality, loss of semantic information learnt, the Grand Tour takes data in 2-Dimensional data and represents it. By simplifying such visualizing into 2-dimensional data at a time, the article argues origin is fixed and with a random velocity, the data points can rotate naturally with the change in distribution of points from the softmax layer. The article finally concluded by explaining Singular Value Decomposition (SVD) theorem from Linear Algebra, which when applied to layers such as Softmax layer, can show skews, transformations and rotations to represent the change in classification of points in multidimensional space without changing the actual data points. This algorithm also uses Eigen space vectors i.e. Eigendecomposition (similar to PCA, still different from that of PCA) for retaining dimensionality of original data in between the transformations applied on the data itself to display animation, without loss of generality. Finally the article shows layer wise, class-wise learning as well class-wise learning across each epoch using Grand Tour. Thus this visualization is certainly a very useful, intriguing example to learn about Image processing data, Neural Networks that learn various types of Images of different numbers of classes. This concept, linear projections, approach can certainly be adapted to Natural Language text.

3 Analysis

3.1 Common Observations from Analysis of Visualization tools

Visualization seems to be well made for Image Processing data, such as for following tools.

1. TensorSpace does not even allow adding RNN or LSTM units from tensorflow - which is a major limitation.

2. Tensorflow Playground - helps if data is fully numerical. Cannot add Recurrent Neural Networks or Long short term memory network or GRU (gated GRU) etc here which constitute Deep learning models' architectures.

3. ANN Visualizer and VisualNN - they create node graphs only showcasing network architecture rather than results through training process or across layers.

4. Tensorboard - which is mainly focused on showing flow charts of the steps of training process, plotting loss, accuracy which can be customized but not extendable to textual corpus training statuses in the network.

5. Quiver_engine - which represents good visualization for Question and answer systems until the software packages it uses were not adapted to newer versions of dependent software packages.

6. Many researchers have explained how BERT works and visualized it. I found one really nice and interesting visualization efforts but specifically towards BERT.https://morioh.com/p/67e7320b3cef

However, as much efforts that did go into Image Processing data visualization over Deep learning models, efforts for Language corpus seems to be far less compared to Image processing dataset.

4 Theory

4.1 Understanding the concept of The Grand Tour

The Grand Tour is a method for viewing multivariate statistical data via orthogonal projections onto a sequence of 2dimensional subspaces. This concept makes use of Grassmannian $Gr(k,V)$ which is a subspace that parameterizes kdimensional linear subspaces of ndimensional subspaces of vector V. For $k = 1$, the Grassmannian $Gr(1,n)$ is the space of lines through the origin in n-space, so it is the same as the projective space of $n-1$ dimensions. [1]

To explain geometrically about Grassmannians: Let vector V consist of $V \cong Kn$ where K is a field. We want to study m dimensional subspaces of V.

Definition:

$$G(m,V)W \subset V, Wsubspace, dim(W) = m \tag{1}$$

.

If $V = Kn$ then we write, $G(m,n)$ for $G(m,V)$, which is to say Linear subspaces in projective space. Thus the Grassmanians will parameterize linear subspaces as follows:

1. $G(m+1,n+1) =$ m planes with $\mathbb{P}m$ inside $\mathbb{P}n$

2. $W \subset \mathbf{K}(n+1)$ then natural projectilization map from $\mathbf{W}$ to $\mathbb{P}\mathbf{W}$

3. And natural projectilization map from $\mathbf{K}(n+1)$ to $\mathbb{P}n$.

4. W is isomorphic to $\mathbf{K}(m+1)$ and $\mathbb{P}\mathbf{W}$ is isomorphic to $\mathbb{P}\mathbf{m}$

There is Duality in this notation, so, $V* =$ dual space of $V =$ linear maps from V to K i.e. $Homlinear(V,K)$. Then $G(m,V) \rightarrow G(n-m,V^*)$

$W \subset V \rightarrow W^+ \subset V^*$ where W^+ with annihilator is a space of linear functions that are identically 0, thus the subspace here will be n-m subspace of linear functions which should reverse map to m dimensional space of W, which formally defines duality (a reverse mapping of defining duality) i.e. to say $(^*)^* = V$.

To visually understand this, this idea of Grassmanian is similar to Gauss Map as follows, where the Gauss map provides a mapping from every point on a curve or a surface to a corresponding point on a unit sphere.

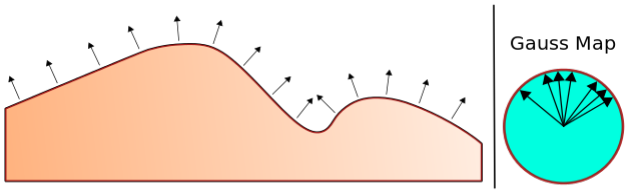

Figure 13: Gauss map example: `https://en.wikipedia.org/wiki/Gauss_map`.

4.2 Image data Visualization

Examining the The Grand Tour work for Image Classification data over Convolutional Neural Network from [6] and the code from which is developed in D3.js and using npm, has helped to understand that the code and r package used is work-in-progress and not completely available as paper is processing of being published. It seems like this is not replicable. Analyzing R's tourr package which is a visualization package that uses grand tour, guided tour concepts (over subspaces and variation of subspaces from Grassmannian), [5] and the published paper [13] , I found that this was an interesting package, however cannot be easily adapted to Neural Networks. There are a lot of BioR packages that use pure multivariate data such as imaging data to understand patterns of proteins and genes in visual space. "tourr: An R Package for Exploring Multivariate Data with Projections" was helpful. Some of the concepts are deeply rooted in Computational Geometry and algebra.

Additionally, examining the [7] and `https://github.com/distillpub/post--grand-tour`, the implementation and design details are yet to be shared by researchers.

4.3 Visualization of LSTM Deep learning model over text data

1. I attempted to visualize LSTM network i.e. Long Short term memory Network for text corpus from RNN visualization based on a tutorial. This is a Google Colab notebook which experiments with most visualization packages. `https://tinyurl.com/4kxm9emn`.

2. This is link to another notebook for medical diagnosis analysis from text data and visualizaiton: `https://tinyurl.com/ysn257v8`

3. One more Notebook for Tourr package for The Grand Tour concept: `https://drive.google.com/file/d/1jbxJisbWn9tW7rVBEaze1l8YyMZiNlgS/view?usp=sharing`

Additionally I created comparison table for the expected Visualization features for Image Processing tasks and Natural language Processing tasks over the visualization tools.

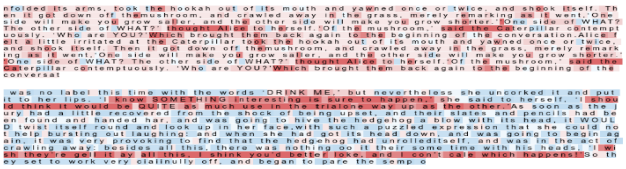

Figure 14: Activations at epoch 169 and epoch 175 during LSTM over Alice in Wonderland book text corpus

	Image data support	Visualize Activations	Visualize Weights	Attributions	Predictions from layer	Predictions from epochs	Unfired neurons vs fired neurons
Tensorflow Playground	No	No	No	No	No	No	
ANN Visualizer	No	No	No	No	No	No	
Simple Neural Network visualizer VisualNN	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Tensorboard	Yes	No	No	No	No	No	
TensorSpace	Yes	Yes	Yes	Yes	Yes	Yes	Yes
quiver_engine	No	No	No	No	No	No	
The Grand Tour	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Figure 15: Taxonomy of Features for Visualization tools over Image Processing tasks. [7]

	Text data support	Visualize Activations	Visualize Weights	Attributions	Predictions from layer	Predictions from epochs	Unfired neurons vs fired neurons
Tensorflow Playground	No	No	No	No	No	No	No
ANN Visualizer	No	No	No	No	No	No	No
Simple Neural Network visualizer VisualNN	No	No	No	No	No	No	No
Tensorboard	NO	No	No	No	No	No	No
TensorSpace	No	No		No	No	No	No
quiver_engine	No	No	No	No	No	No	No

Figure 16: Taxonomy of Features for Visualization tools over Natural Language Processing tasks. [7]

4.4 UMAP and t-SNE

T-sne algorithm [12]

1. Tsne is tdistributed stochastic neighbor embedding, where the method makes use of Student's tdistribution to visualize high dimensional data by giving each data point a location in a two or three dimensional map.

2. This method is used for nonlinear dimensionality reduction. Example: Natural Language processing corpus to represent word embeddings i.e. to demonstrate the distributional hypothesis from linguistics over word embeddings (distances between words).

3. Thus tsne makes sure data points that are closer are preserved and hence tend to occur more near in tsne embeddings.

4. It seems the distances between pair of points is calculated and nearest distances are assigned higher probability and hence a probability distribution over pairs of high dimensional data objects is constructed by tsne.

5. Dissimilar distances between pairs of data points are assigned lower probability in low-dimensional maps.

6. Kullback − Leibler divergence which is called as KL divergence is used to minimize the KL divergence between the two probability distributions wŕt locations assigned in the maps.

7. The key steps involve:

   (a) Computing probabilities:

   $$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2/2\sigma^2)}{\sum_{k\neq i}\exp(-\|x_i - x_k\|^2/2\sigma^2)} \quad (2)$$

   (b) Learning goal of tsne to learn n-dimensional map Y that reflects pij as well as possible which sets $q_{ii} = 0$ which makes it a heavy tailed tdistribution and thus ensures dissimilar points to be far apart:

   $$q_{ij} = \frac{(1+\|y_i - y_j\|^2)^{-1}}{\sum_{k\neq l}(1+\|y_i - y_l\|^2)^{-1}} \quad (3)$$

   (c) compute gradient:

   $$\frac{\delta C}{\delta y_i} = 4\sum_j (p_{ij} - q_{ij})(y_i - y_j)(1+\|y_i - y_j\|^2)^{-1} \quad (4)$$

   (d) The KL divergence minimization condition, which uses Gradient descent algorithm for solving minimization problems.

   $$KL(P||Q) = \sum p(X)\log(p(X) \div q(X)) \quad (5)$$

   (e) The algorithm uses the Gaussian kernel over euclidean distance between a pair of points. Since Euclidean distance introduces the curse of dimensionality, so the tsne algorithm used finally adjusts the problem of discriminating the distance in higher dimensions.

UMAP [8]

1. UMAP uses exponential probability distribution in high dimensions but not necessarily Euclidean distances like tSNE but rather any distance can be plugged in. In addition, the probabilities are not normalized.

2. Visually, it is similar to tSNE, but it assumes that the data is uniformly distributed on a locally connected Riemannian manifold and that the Riemannian metric is locally constant or approximately locally constant. [3] [8]

5 Conclusion & Reflection

Generally, given the complexity of Neural networks, it can be difficult to visualize as per user's requirement. However text visualization is difficult to visualize given there many aspects to consider as per analysis from Image data.

Summary of analysis is as follows: I wrote down and attempted to elaborate in simpler description here: https://tinyurl.com/3w5n2ve7

1. Considering the above analysis, it seems Neural networks and Deep learning models are not too familiar for the class. This added to the complexity of problem description. However, I attempted to write down and redefine and narrow the goal down, as per guidance from professor, so much further so my main focus is now on following:

(a) how to explain the already complex problem to an audience who may not be familiar with neural networks/Deep learning?

(b) how to describe language processing tasks to an audience who may not be familiar with neural networks/Deep learning?

(c) To explain visually, requires not only the approach to explain to human audience, but also to simplify the visualization approaches that a DNN can represent and provide without loss of information during visual transformations over Text corpus.

(d) Additionally, to follow through this task, I need to spend some more time, similar to how I learnt about above concepts and works during weekly journal task, I could narrow the problem further by a simple example and define what specific goals I want to achieve and how to visualize such a small example similar to [7].

6 Future work

I would be continuing to work on developing the concept to visually explain Neural network over text corpus in a similar fashion from Image processing examples from above examples and analysis, by reaching out to researchers and as well with possibly with continued work with professor. I would like to apply the intersection of First Order Logic and the Q&A for Language Understanding tasks and would like to visualize over a small dataset. [2] This can be done from a basic Recurrent Neural Network Unit at first and elaborate and expand the model to analyze and explain from more advanced NLP DNNs. I would like to learn the Singular Vector Decomposition and Linear projects described in [7] for Image processing data.

7 Project Artifacts

1. **Project Powerpoint:** `https://tinyurl.com/pyc5rf6z`
2. **Project Video:** `https://tinyurl.com/2bvmzm6u`
3. **Google Colab Notebook analysis of Visualization tools:** `https://tinyurl.com/4kxm9emn`
4. **Medical Diagnosis data visualization:** `https://tinyurl.com/ysn257v8`
5. **Tourr package for The Grand Tour concept:** `https://tinyurl.com/5n86efrn`

Acknowledgments

I would like to thank Prof. Abram Handle for guidance, encouragement and support. This work was part of course at University of Colorado Boulder, Information Visualization 5602 Graduate level course.

References

[1] D. Asimov. The grand tour, siam jrnl. on sci. and stat. *Computing*, 6(1):128f, 1985.

[2] G. Ciravegna, F. Giannini, M. Gori, M. Maggini, and S. Melacci. Human-driven fol explanations of deep learning. In *IJCAI*, pp. 2234–2240, 2020.

[3] A. Coenen and A. Pearce. Understanding umap. *Google PAIR*, 2019.

[4] K. Kucher and A. Kerren. Text visualization techniques: Taxonomy, visual survey, and community insights. In *2015 IEEE Pacific visualization symposium (pacificVis)*, pp. 117–121. IEEE, 2015.

[5] U. Laa.

[6] M. Li, Z. Zhao, and C. Scheidegger.

[7] M. Li, Z. Zhao, and C. Scheidegger. Visualizing neural networks with the grand tour. *Distill*, 5(3):e25, 2020.

[8] L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[9] G. Ras, M. van Gerven, and P. Haselager. Explanation methods in deep learning: Users, values, concerns and challenges. In *Explainable and interpretable models in computer vision and machine learning*, pp. 19–36. Springer, 2018.

[10] G. Stanovsky and I. Dagan. Creating a large benchmark for open information extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2300–2305, 2016.

[11] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[12] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[13] H. Wickham, D. Cook, H. Hofmann, and A. Buja. tourr: An r package for exploring multivariate data with projections. *Journal of Statistical Software*, 40(2):1–18, 2011. doi: 10.18637/jss.v040.i02