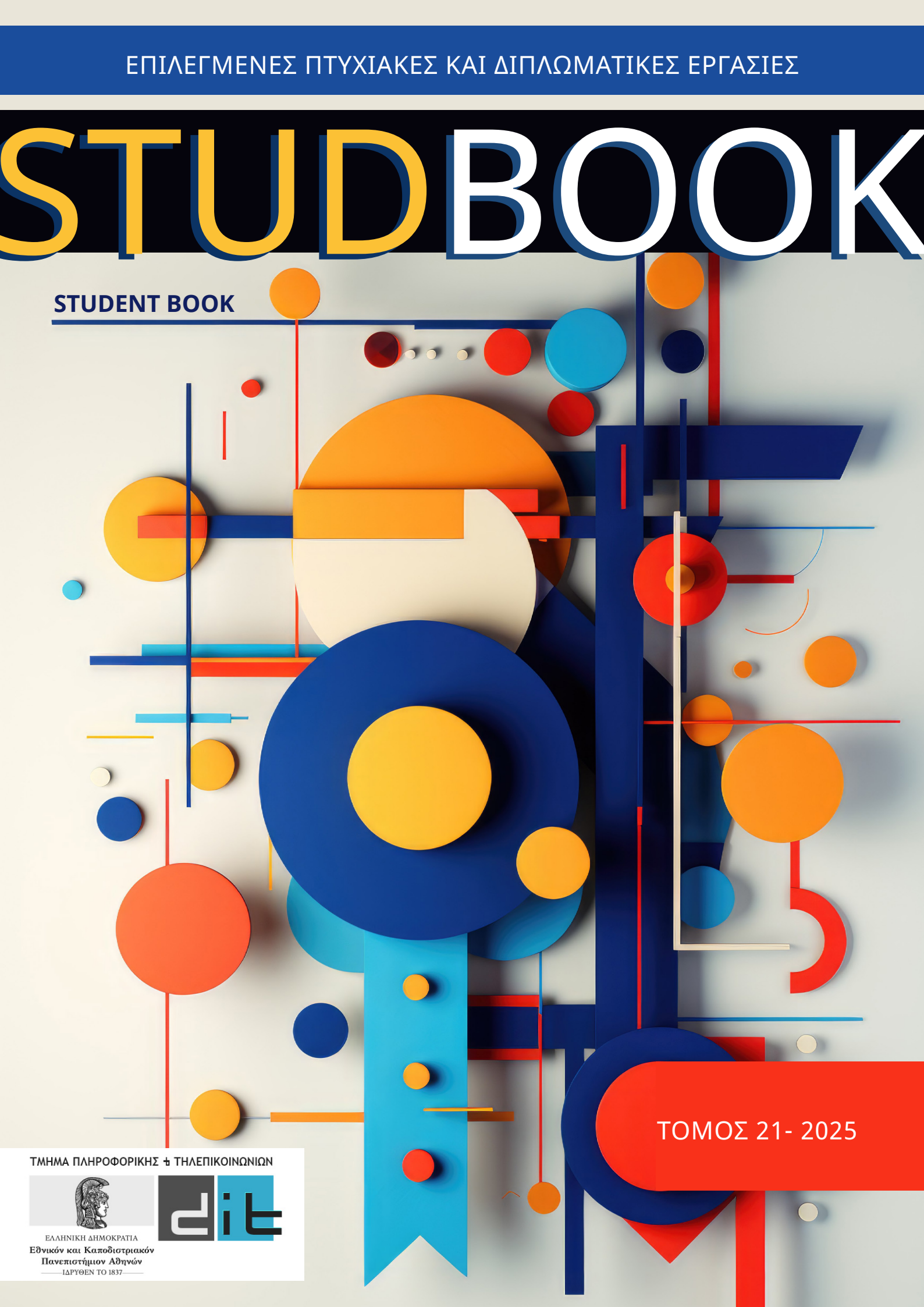
ΕΠΙΛΕΓΜΕΝΕΣ ΠΤΥΧΙΑΚΕΣ ΚΑΙ ΔΙΠΛΩΜΑΤΙΚΕΣ ΕΡΓΑΣΙΕΣ

STUDBOOK

STUDENT BOOK



ΤΟΜΟΣ 21- 2025

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικόν και Καποδιστριακόν Πανεπιστήμιον Αθηνών
ΙΔΡΥΘΕΝ ΤΟ 1837

Εκδίδεται μία φορά το χρόνο από το:

Τμήμα Πληροφορικής και Τηλεπικοινωνιών
Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών,
Πανεπιστημιούπολη, 15784 Αθήνα

Επιμέλεια έκδοσης:

Χάρης Θεοχάρης (Καθηγητής), Υπεύθυνος Έκδοσης

Λήδα Χαλάτση (ΕΤΕΠ), Σύνταξη & Γραφιστική Επιμέλεια

Εικόνα εξωφύλλου και εσωφύλλων:
Vecteezy (https://www.vecteezy.com/free-photos/abstract)

ISSN
1792-8826

Περιεχόμενα

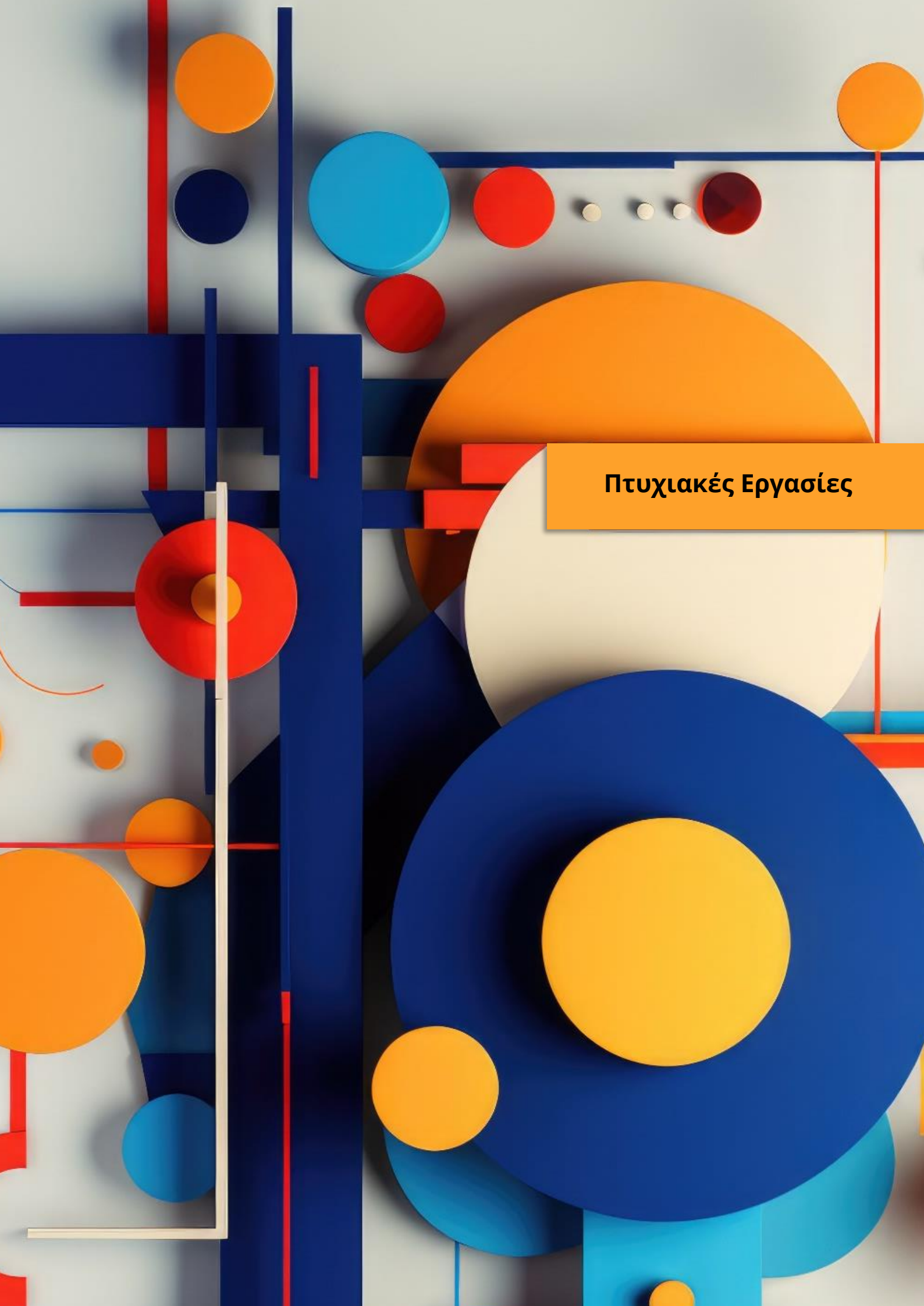
Πρόλογος

Ο τόμος αυτός περιλαμβάνει περιλήψεις επιλεγμένων διπλωματικών και πτυχιακών εργασιών που εκπονήθηκαν στο Τμήμα Πληροφορικής και Τηλεπικοινωνιών του Εθνικού και Καποδιστριακού Πανεπιστημίου Αθηνών κατά το διάστημα **01/01/2024 - 31/12/2024**. Πρόκειται για τον **21ο τόμο** στη σειρά αυτή. Στόχος του θεσμού είναι η ενθάρρυνση της δημιουργικής προσπάθειας και η προβολή των πρωτότυπων εργασιών των φοιτητών του Τμήματος.

Η έκδοση αυτή είναι ψηφιακή, έχει δικό της ISSN και αναρτάται στην επίσημη ιστοσελίδα του Τμήματος ώστε να έχει μεγάλη προσβασιμότητα. Για το στόχο αυτό, σημαντική ήταν η συμβολή της Λήδας Χαλάτση που επιμελήθηκε και φέτος την ψηφιακή έκδοση και πέτυχε μια ελκυστική ποιότητα παρουσίασης, ενώ βελτίωσε και την ομοιογένεια των κειμένων.

Θα θέλαμε να ευχαριστήσουμε τους φοιτητές για το χρόνο που αφιέρωσαν για να παρουσιάσουν τη δουλειά τους στα πλαίσια αυτού του θεσμού και να τους συγχαρούμε για την ποιότητα των εργασιών τους. Ελπίζουμε η διαδικασία αυτή να προσέφερε και στους ίδιους μια εμπειρία που θα τους βοηθήσει στη συνέχεια των σπουδών τους ή της επαγγελματικής τους σταδιοδρομίας.

Αθήνα, Ιούνιος 2025



Πτυχιακές Εργασίες

Short-term Wildfire Danger Forecasting Methods in the Mediterranean Using Deep Learning

Eleftheria Vrachoriti

ABSTRACT

The intensity of extreme weather conditions responsible for frequent wildfire ignitions in the Mediterranean has been exacerbated due to climate change. Deep Learning (DL) models are capable of accurately predicting the daily wildfire danger of a region based on satellite-derived and meteorological data. Nevertheless, predicting the wildfire risk for longer periods of time is quite challenging. In this Thesis, we attempt to combine multi-step forecasting strategies with DL models to extend the forecast horizon up to 10 days past the day of the last observation. To assess each method, we conduct a comparative study at different forecast horizons and also an extensive analysis to determine the optimum window length for each prediction step. Our results indicate that the time series used for training can be reduced to at least 25-40% of its original length, without compromising prediction quality. An ablation study on feature groups of different sources and types shows that the satellite-derived features are essential for making accurate predictions. Finally, we employ Explainable Artificial Intelligence (xAI) techniques to shed light on the inherent mechanisms of wildfire danger forecasting models of different forecasting window lengths or horizons. Our findings suggest that the last 2-3 days prior to fire ignition are the most important when it comes to making multi-step predictions.

Keywords: Machine Learning, Deep Learning, Wildfire Danger, Forecasting Methods, Explainable Artificial Intelligence

ADVISORS

Manolis Koubarakis, Professor, National Kapodistrian University of Athens
Ioannis Papoutsis, Assistant Professor, National Technical University of Athens

1 INTRODUCTION

Almost half (48.7%) of extreme wildfires in the Mediterranean are heat-induced [1]. Climate change is expected to increase the frequency and intensity of fire weather conditions in the Mediterranean [1, 2], potentially increasing the risk of wildfire ignition. This underlines the need for developing intelligent systems that provide reliable predictions of wildfire hazard to improve fire management strategies.

DL methods have been very successful in predicting and managing wildfires, making them more popular in the last decade [3]. However, most of the work in this field focuses on predicting the next day's wildfire danger. In this work, we combine multi-step forecasting methods with DL to extend the forecast horizon up to 10 days beyond the last observation. Then, models with different forecast horizons are compared using a common predefined window length. An ablation study is then carried out to determine the optimal window length for each forecast horizon, attempting to minimize the forecast window with little or no loss of performance.

Despite the great performance of DL methods, they are often hard to interpret due to their black-box nature [4]. Yet, unveiling the relations between fire drivers, as they have been modeled at the end of training, could enhance the comprehension of fire ignition conditions and help us build more efficient models. In this context, we shed light on the importance of different fire drivers in both next day’s wildfire danger prediction models, as well as models of different forecast horizons using various xAI methods.

2 BACKGROUND AND RELATED WORK

Conventionally, the Fire Weather Index (FWI) [5] is used to estimate the next day’s fire danger. However, its calculation is based on meteorological data and completely disregards other wildfire drivers, such as the local topography or land coverage information of the area or human activity indicators.

On the other hand, ML methods are pretty accurate in predicting the occurrence and severity of extreme events, such as wildfires. DL techniques extract features from large datasets of multimodal data and exploit both the spatial and temporal context of wildfires, often surpassing other ML techniques in wildfire related tasks.

Prapas et al. [6] identified the challenges of treating the next day's wildfire danger prediction as an ML task and trained models on historical data of wildfires in Greece, based on different contexts; pixel, spatial, temporal and spatio-temporal. Their findings suggest that exploiting the spatial or temporal context of wildfires improves accuracy, while the best results are yielded when both spatial and temporal contexts are used.

Kondylatos et al. [7] extended the area of interest to the Eastern Mediterranean, training the same models on datasets of pixel, temporal and spatio-temporal context. Results showed that the DL methods were more accurate than FWI. They also applied various xAI methods to investigate the effect of different features on specific severe wildfire evens, to examine the average impact of the most conventionally important drivers on the model's prediction, and to investigate the temporal evolution of the integrated gradients of the model the days preceding the ignition. Overall, their findings indicated that models tend to base their predictions mostly on fuel-related features.

Building on this, Kondylatos et al. [8, 9] published Mesogeos, the largest available dataset of historical ignition points and burned area sizes of wildfires in the Mediterranean, which includes a wide range of both dynamic and static fire covariates. They also published baseline models, including an LSTM, a Transformer, a Gated Transformer Network (GTN).

However, aforementioned works focus on daily wildfire danger forecasting. While predictions for shorter forecast horizons are more accurate and contain less uncertainty, acquiring the fire risk for longer time periods would be invaluable for improving forestry management and wildfire mitigation strategies in fire-prone areas, such as the Mediterranean. In this direction, we extend the forecast horizon up to 10 days by applying multi-step forecasting methods with DL models. An evaluation of the performance of all methods in all prediction

steps is carried out using a common predefined forecast window length. Then, an ablation study on the forecast window length takes place, to determine the shortest possible window length that does not compromise prediction quality. Additionally, we ablate different groups of features to determine their importance in multi-step predictions. Lastly, we extend the application of xAI methods in [7] to the whole Mediterranean region, to determine how the focus of the models differentiates as the forecast horizon shrinks or as the number of observations decreases.

3 DATA AND METHODS

3.1 Data

The data used in this Thesis is extracted from the Mesogeos datacube [8, 9], and is used for training DL models for wildfire danger forecasting. This dataset consists of 8547 positive and 17342 negative samples and contains 35 variables, out of which 24 are used for training. The input variables are either dynamic or static. The dynamic variables are either sourced of meteorological stations or are satellite-derived, while the static variables model indicators of human activity and local topography or land cover.

The forecasting methods that we have employed utilize the temporal context of the data, i.e. each sample is a time series of 30 days preceding a fire event.

3.2 Models

The models used in this thesis are the same as those mentioned in the Mesogeos paper [8]; a Long Short-Term Memory (LSTM), a Transformer and a Gated Transformer Network (GTN) that uses the attention mechanism both in time and feature dimensions.

3.3 Loss functions

In next day forecasting and direct forecasting, the end goal is to classify each sample correctly, so the Cross Entropy (CE) loss function is used [8], while in iterative forecasting, the Mean Squared Error (MSE) loss function is also used to monitor the forecasting loss.

3.4 Metrics

The same metrics as in [8] are used to compare all models; accuracy, precision, recall, f1-score and Area Under Precision-Recall Curve (AUPRC).

3.5 Time Series Forecasting

3.5.1 Forecasting

Forecasting refers to the practice of predicting the values of a variable y in the future based on past observations. The set of past observations Y used for forecasting is called forecast window. The number of observations L used is called forecast window length and is also often referred to as lag. The last available observation y_t is called forecast origin. The number of steps predicted into the future H is called forecast horizon [10].

3.5.2 Forecasting Methods and Strategies

Forecasting methods are divided into subcategories based on the number of variables that are forecasted as well as on the length of the forecast horizon.

3.5.2.1 Univariate and Forecasting

Univariate forecasting methods predict the future values of a single variable y. In this case, the forecasting window Y is a vector consisting of L past observations of this particular variable y:

$$Y = [y_{t-L+1}, \dots, y_{t-1}, y_t] \quad (3.1)$$

3.5.2.2 Multivariate Forecasting

Multivariate forecasting methods predict the future values of a set of M variables $y_1, y_2, \dots, y_M$ at the same time. In this case, the forecasting window Y is a matrix of size $M \times L$ consisting of row vectors Y_i that contain L past observations for each variable y_i, $i = 1, \dots, M$ [11]:

$$Y = \begin{bmatrix} Y_1 \\ \dots \\ Y_M \end{bmatrix} = \begin{bmatrix} y_{1,t-L+1} & \cdots & y_{1,t-1} & y_{1,t} \\ \cdots & \cdots & \cdots & \cdots \\ y_{M,t-L+1} & \cdots & y_{M,t-1} & y_{M,t} \end{bmatrix} \qquad (3.2)$$

3.5.2.3 One-step Forecasting

In univariate forecasting, one-step methods are used to forecast a single variable y one step into the future, i.e. given the forecasting window Y defined in (3.1) to predict y_{t+H} where $H = 1$. Similarly, in multivariate forecasting, the above definition is extended to forecasting a set of variables $\{y_i\}$, $i = 1, \dots, M$ one step into the future, i.e. given the forecasting window defined in (3.2) to predict $y_{i,t+H}$, where $H = 1$.

3.5.2.4 Multi-step Forecasting

Multi-step forecasting methods are a generalization of one-step forecasting methods for a larger number of steps. In this case, the forecast horizon is extended and all next $H > 1$ values are calculated.

In univariate forecasting, given the forecasting window Y defined in (3.1), the values $y_{t+1}, y_{t+2}, \dots, y_{t+H}$, are predicted for $H > 1$ [12]. In the multivariate case, given the forecasting window defined in (3.2), the values $y_{i,t+H}$, are predicted for $H > 1$ [11].

3.5.2.4.1 Iterative Forecasting

The iterative forecasting strategy [12] iteratively uses a single one-step forecaster f to produce all predictions until the forecast horizon H is reached. After each step, the predictions are appended to the forecast window and the oldest observations are removed, to ensure lag remains unchanged. In multivariate forecasting, this scheme is extended by using a one-step multi-input multi-output forecaster f. This can also be described in the following pseudocode:

Algorithm 1 Iterative Forecasting

Input: one-step forecaster $f: \mathbb{R}^{M\times L} \to \mathbb{R}$, forecast window $Y = [Y_{t-L+1}, \dots, Y_t] \in \mathbb{R}^{M\times L}$ (in univariate forecasting, row vector Y_i degrades to y_i) and forecast horizon $H > 1$

Output: forecasts $\hat{Y} = [\hat{Y}_{t+1}, \dots, \hat{Y}_{t+H}] \in \mathbb{R}^{M\times H}$

1: $Y = \emptyset$
2: **for** $h = 1$ to H **do**:
3: $\quad \hat{Y}_{t+h} = f(Y)$
4: $\quad \hat{Y} = \hat{Y} \cup \hat{Y}_{t+h}$
5: $\quad Y = [Y_{t+L-1}, \dots, Y_t, \hat{Y}_{t+1}, \dots, \hat{Y}_{t+h-1}]$
6: **end for**
7: **return** $\hat{Y}$ as an $M \times H$ matrix

The iterative forecasting strategy, while being intuitive and not requiring a lot of computational resources, it has a major downside; as the number of steps increases, observations are replaced by forecasts leading to the accumulation of prediction error.

3.5.2.4.2 Direct Forecasting

The direct forecasting strategy [12] uses H independent forecasters f_h, each producing predictions at a particular step h, $h = 1, \dots, H$. In this case, the forecast window is common for all models and remains unchanged during the whole forecasting process. The direct forecasting method can also be described with the following pseudocode:

Algorithm 2 Direct Forecasting

Input: H independent one-step forecasters $f_h: \mathbb{R}^{M\times L} \to \mathbb{R}$, $h = 1, \dots, H$, forecast window $Y = [Y_{t-L+1}, \dots, Y_t] \in \mathbb{R}^{M\times L}$ (in univariate forecasting, Y_i degrades to y_i) and forecast horizon $H > 1$

Output: forecasts $\hat{Y} = [\hat{Y}_{t+1}, \dots, \hat{Y}_{t+H}] \in \mathbb{R}^{M\times H}$

1: $Y = \emptyset$
2: **for** $h = 1$ to H **do**:
3: $\hat{Y}_{t+h} = f_h(Y)$
4: $\hat{Y} = \hat{Y} \cup \hat{Y}_{t+h}$
5: **end for**
6: **return** $\hat{Y}$ as an $M \times H$ matrix

The direct forecasting strategy is more robust to model misspecification, as the prediction error at a particular step h does not influence the performance of the rest of forecasters. However, it is quite expensive, since it requires training H different models.

3.5.3 Time Series Forecasting Using Machine Learning

In univariate forecasting, one-step models produce a single output that corresponds to the prediction of the forecasted variable. Artificial Neural Networks (ANNs) with a single output node can be used for this purpose. In multi-step forecasting, we can either use this model as a one-step forecaster in the direct or iterative forecasting strategy, or add H output nodes to produce all H forecasts at once [11].

In multivariate forecasting, both one-step and multi-step methods ought to have a MIMO structure to produce forecasts for all variables at once in the form of an M-sized vector or an $M \times H$ matrix. Similar to univariate forecasting, multivariate one-step models can be used as one-step forecasters in either of the aforementioned strategies. Possible ML models for multivariate forecasting are ANNs with multiple nodes at the output layer, or more sophisticated architectures that exploit the temporal context of the data, such as RNNs, LSTMs or Gated Recurrent Units (GRUs).

3.6 Explainable Artificial Intelligence (xAI)

3.6.1 Partial Dependence Plots (PDPs)

Partial Dependence Plots (PDPs) are an xAI method that visualizes the marginal effect of a feature on the model's prediction [13, 14]. Let f be the output function of the model and X be the set of all variables $X = \{x_1, x_2, \dots, x_M\}$. Let $Z \subset X$ be a set of variables of interest, $z_i \in X$, $i = 1,2, \dots, P$and C be the complimentary set of variables, i.e. $Z \cup C = X$, then the partial dependence function f_Z of f on Z is defined as follows [14]:

$$f_Z = E_C[f(Z, C)] = \int f(Z, C)p(C)dC \qquad (3.3)$$

where each subset Z has its own dependence function f_Z, which gives the average value of f when Z is fixed and C varies over its marginal distribution Z.

f_Z can then be estimated using the following sum:

$$\hat{f}_Z = \sum_{i=1}^{N} f(Z, C_i) \qquad (3.4)$$

where C_i represents the different permutations of values of variables in C.

3.6.2 Partial Dependence Plots (PDPs)

Integrated Gradients (IGs) is an xAI method that is applied to DL models to compute feature attributions. Suppose we have function $f: \mathbb{R}_n \to [0,1]$ representing a Deep Neural Network (DNN). Let $x \in \mathbb{R}_n$ be the input at hand and $x' \in \mathbb{R}_n$ be the baseline input, then we consider the straight-line path (in $\mathbb{R}_n$) from x' to x and compute the integral of the gradients at all points along the path [15]:

$$IG_i(x) = (x_i - x_i') \int_{\alpha=0}^{1} \frac{\partial f(x' + \alpha(x - x'))}{\partial x_i} \partial\alpha \qquad (3.5)$$

The integral in (3.5) can then be approximated via the following summation:

$$IG_i^{approximate}(x) = (x_i - x_i') \sum_{k=1}^{m} \frac{1}{m} \frac{\partial f\left(x' + \frac{k}{m}(x - x')\right)}{\partial x_i} \qquad (3.19)$$

where m is the number of steps in the Riemann approximation of the integral.

4 EXPERIMENTS

4.1 Forecasting

4.1.1 Iterative/Recursive Forecasting

To apply the H-step iterative forecasting strategy to our task, we first train a forecast LSTM layer with 128 neurons, that, given a time series with $30 - H$ days lag, predicts the feature values the $(30 - H + 1)$-th day. The trained model is then used as a one-step forecaster in Algorithm 1 to forecast the variables in remaining days $T + 1$, $T + 2$, etc., until the 30th day, $T + H$, is reached. Then, we concatenate the initial time series with the forecasts of each intermediate step to construct a time-series with 30 days lag, which can be fed into any of the three baseline models to perform wildfire danger forecasting.

4.1.2 Direct Forecasting

To apply the H-step direct forecasting strategy, models with the same architecture as the three baselines are trained but with $30 - H$ days lag, as described in Algorithm 2. In this case, we do not perform any predictions on the features, but on the wildfire danger of the 30th day.

4.1.3 Comparison of Different Forecasting Methods

As the number of steps and lag are changing at the same time, making comparisons of performance at different steps harder, because the effect of each of the two factors on the performance is not clear. Therefore, we carry out the training and testing process using a constant lag of 20 days for all methods and steps.

4.1.4 Ablation Study Based on Lag

Apart from the experiments described earlier, we perform an ablation study to determine the optimum lag for each method and step. For H-step forecasting, we apply the respective method with all possible lags L, $L = 1,2, \dots ,30 - H$ to see how the performance is affected.

4.1.5 Feature Ablation in Groups

It is useful to experiment with ablating features from our models to observe how the next day with 20 days lag and 1, 5, 10-step forecasting performance with 20 days lag is affected. In this series of experiments, we focus on the LSTM baseline model.

We begin by removing the satellite-derived data from the model, so that it solely relies on meteorological and static data for its predictions. Then, we gradually ablate the static features to determine their significance in achieving a high prediction score. Since the number of such features is relatively large, we avoid ablating them one by one. Instead, we sort them into groups, according to and ablate all combinations of groups; first, each group separately, then pairs of groups and lastly, all groups, completely removing the static data. In each ablation, the model is trained from scratch.

The results of these experiments are indicative of the importance of each feature group. If ablating a group leads to worse performance than that of the baseline, then it is important, otherwise it is not valuable for our task and/or model. Additionally, we can sort different groups by their importance. For example, if ablation of group A leads to worse performance than of group B, then A is of greater importance than B.

4.2 xAI

4.2.1 Partial Dependence Plots (PDPs)

We produce the PDPs for an LSTM trained for 1, 5 and 10-step Direct forecasting with 20 days lag, to find out if the most important features change when the forecast horizon increases.

4.2.2 Integrated Gradients (IGs)

We produce the IG plots for an LSTM trained for 1, 5 and 10-step Direct forecasting with 20 days lag to investigate how changes in lag and forecast horizon affect the temporal evolution of different fire drivers.

5 RESULTS AND DISCUSSION

5.1 Forecasting

5.1.1 Iterative Forecasting

In iterative forecasting, the forecasting loss is high even at the very first step and it accumulates as the number of steps increases (Figure 5.1) as expected, which has a great impact on the classification test loss, leading to very poor performance.

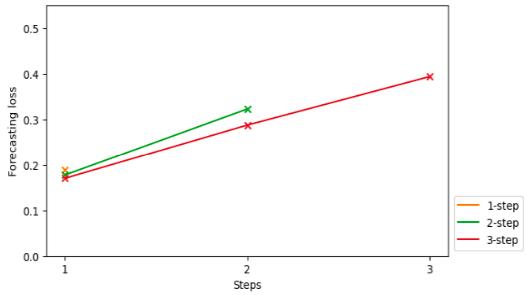
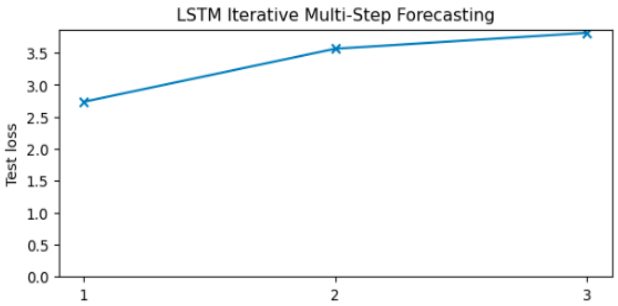


Fig. 5.1. Comparison of 1, 2 and 3-step Iterative forecasting using the LSTM forecaster and the LSTM baseline classifier with 20 days lag, based on (a) the forecasting loss and (b) classification loss

5.1.2 Direct Forecasting

The loss increases as the forecast horizon increases, but the largest increase in loss takes place during the first 5 steps. The rest of metrics are not affected as much, and in fact, even 10-step forecasting yields acceptable results considering the long forecast horizon.



Even though all three models are pretty much indistinguishable in terms of performance (Figure 5.2), the most stable model for smaller forecast horizons seems

to be the LSTM, while for larger horizons the GTN is probably a better choice.

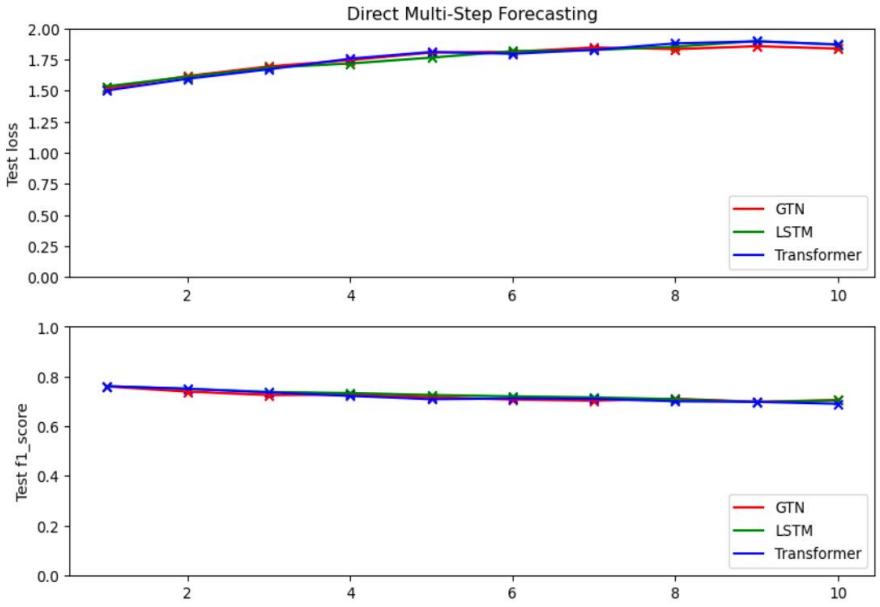


Fig. 5.2. Comparison of 1, 2, ..., 10-step Direct forecasting using the LSTM, GTN and Transformer baselines with 20 days lag based on classification metrics

5.1.3 Ablation Study Based on Lag

In iterative forecasting, the forecasting loss increases as the lag decreases, as seen in Figure 5.3, however the classification metrics do not follow a specific increasing or decreasing trend but are quite unstable, as they are oscillating in a range of 3 to 5%.

Regarding the direct forecasting strategy, for 1, 2, 3 and 4-step forecasting, the LSTM baseline is very robust. Even when using just a week of historical data, which corresponds to a reduction of the original maximum possible lag by 73-76%, the results produced are still comparable, while the performance gradually gets worse for lags smaller than a week. Furthermore, when the lag is longer or equal to 3 days, the following relation can be observed for two consecutive models using the same lag:

$$e_h^l \leq e_{h+1}^l,\ l = 3, \ldots, 30 - h,\ h = 1,2,3 \qquad (5.1)$$

For 5-step forecasting, it is probably best to have more than 10 days of historical data for better results, which corresponds to a roughly 60% reduction on the original lag. However, for longer forecast horizons, the results seem very unstable (Figure 5.4), making it almost impossible to come to a conclusion about the optimum lag.

Fig. 5.3. Comparison of 1, 2 and 3-step iterative forecasting using the LSTM forecaster and LSTM baseline classifier with different lags, based on (a) the forecasting loss and (b) classification metrics

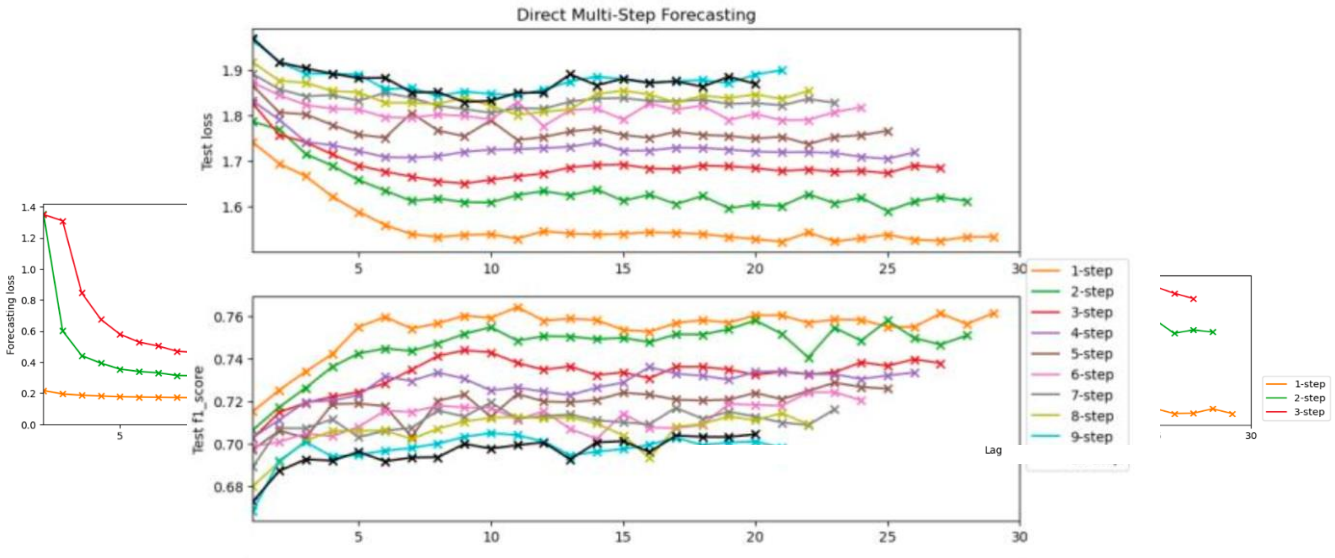


Fig. 5.4. Comparison of (a) 1, 2, ..., 10-step direct forecasting using the LSTM baseline with different lags, based on classification metrics

5.1.4] Feature Ablation in Groups

Removing the satellite-derived in 1, 5 and 10-step direct forecasting with 20 days lag, leads to worse scores as the forecast horizon expands, indicating that the importance of the satellite-derived data is also increasing.

Regarding the ablation of static data in groups in direct forecasting using the same lag, it is inferred that ablating any of the three groups makes the model less efficient and each group of features has greater importance for larger forecast horizons. The following relations are observed for the classification loss of 1, 5 and 10-step direct forecasting, which underlines the importance of land cover features for all horizons:

$$loss^{h}_{human\_indicators} < loss^{h}_{topography} < loss^{h}_{landcover}, h = 1,5 \quad (5.1)$$
$$loss^{h}_{topography} < loss^{h}_{human\_indicators} < loss^{h}_{landcover}, h = 10$$

Ablating the static data in pairs of groups always leads to worse performance. Additionally, the following relations for the classification loss of 1, 5 and 10-step direct forecasting is observed, which leads to the same conclusion about land cover data:

$$loss^{h}_{human\_ind,topog} < loss^{h}_{topog,land} < loss^{h}_{human\_ind,land}, h = 1 \quad (5.2)$$

$$loss^{h}_{human\_ind,topog} < loss^{h}_{human\_ind,land} < loss^{h}_{topog,land}, h = 5,10$$

Removing all static data completely worsens the classification performance only in 5 and 10-step forecasting, compared to ablating the features in pairs of groups.

5.2 5.2. xAI

5.2.1 LSTM 1, 5 and 10-step Direct Forecasting with 20 Days Lag

Regarding the PDP plots, the lst_day feature seems to be pretty important as it can increase the wildfire danger from almost 0 to 60%.

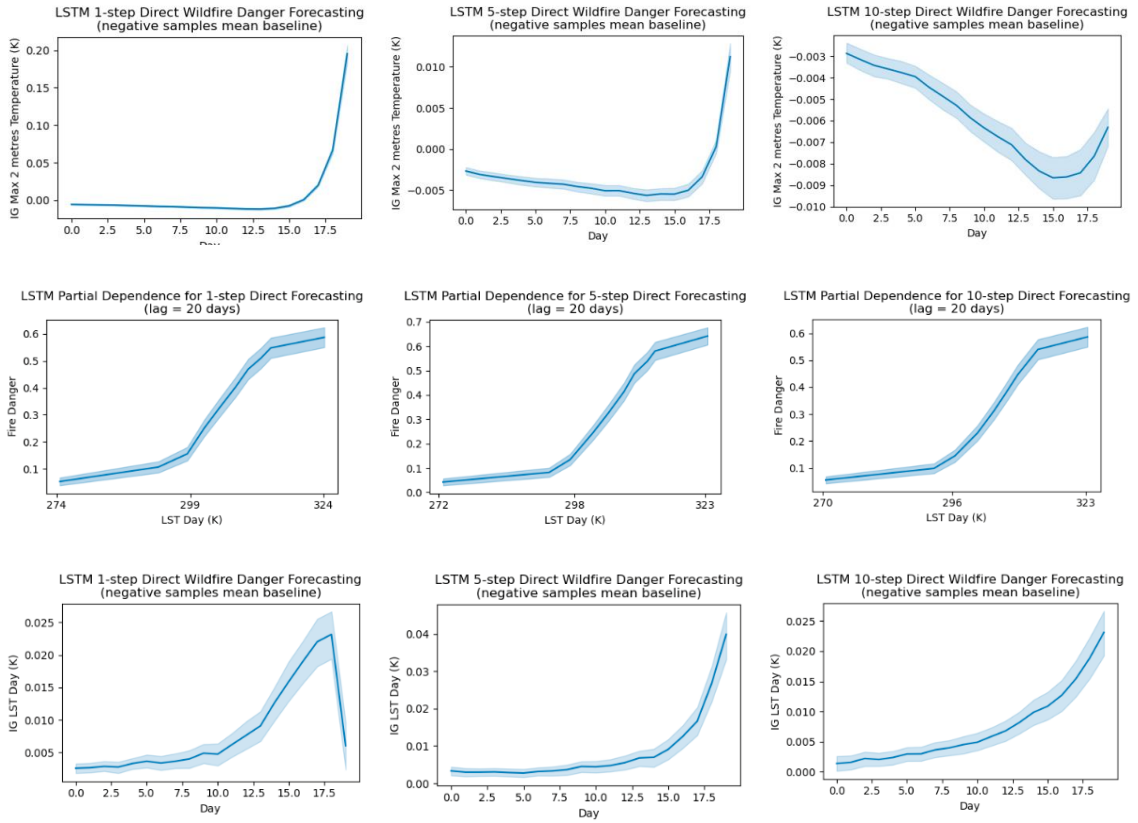


Fig. 5.5. PDP and IG plots for 1, 5 and 10-step direct forecasting using 20 days lag

In the IG plots, in 1-step forecasting, lst_day also appears to be less important than t2m, while the opposite is observed for 5 and 10-step forecasting. Thereby, the model probably relies more on meteorological data for longer forecast horizons.

6 CONCLUSIONS

In this work, we approached the multi-step wildfire forecasting task using the iterative and direct forecasting strategies. The findings suggest that the iterative method is very erroneous and would be better avoided for our problem. In contrast, the direct forecasting strategy yields much better results, though it is still prone to error for very small lags.

The ablation study based on lag showed in iterative forecasting, the optimum lag is at least 5 days, though, the error accumulation is very high. In direct forecasting

with less than 4 steps ahead, the optimum lag is at least 7 days, which is roughly 25% of the original lag, while for 5-step forecasting, the ideal lag is at least 10 days, 40% of the initial lag.

The results of the ablation study based on satellite-derived and static data for 1, 5 and 10-step direct forecasting using the LSTM, indicates tthat the satellite-derived data are of great importance and the most important static features are the land cover data for all steps.

Regarding the xAI methods, PDPs suggest that the lst_day is the most important wildfire covariate for 1, 5 and 10-step direct forecasting using the LSTM, while the IGs plots show that the last 2 or 3 days before the wildfire ignition are the most important days, regardless of lag or forecast horizon and that the LSTM sometimes relies more on satellite-derived data than on meteorological data for its predictions.

7 FUTURE WORK

There is still room for improvement for the iterative forecasting method, by introducing various error correction methods to ensure more accurate predictions of feature values. However, this method is still the most error prone by definition.

Concerning the xAI techniques, we might want to try out more sophisticated xAI methods, specifically designed for time series models to analyze the behavior of the models even more.

REFERENCES

[1] J. Ruffault, T. Curt, V. Moron, R. M. Trigo, F. Mouillot, N. Koutsias, F. Pimont, N. Martin-StPaul, R. Barbero, J. – L. Dupuy, A. Russo, and C. Belhadj-Khedher.

Increased likelihood of heat-induced large wildfires in the Mediterranean Basin. Scientific Reports, vol. 10, no. 1, p. 13790, Aug. 2020.

[2] M. Turco, J. J. Rosa-Cánovas, J. Bedia, S. Jerez, J. P. Montávez, M. C. Llasat, and A. Provenzale. Exacerbated fires in Mediterranean Europe due to anthropogenic warming projected with nonstationary climate-fire models. Nature Communications, vol. 9, no. 1, p. 3821, Oct. 2018.

[3] P. Jain, S. C. P. Coogan, S. G. Subramanian, M. Crowley, S. Taylor, and M. D. Flannigan. A review of machine learning applications in wildfire science and management. Environmental Reviews, vol. 28, no. 4, pp. 478–505, Dec. 2020.

[4] Y. Zhang, P. Tino, A. Leonardis, and K. Tang. A Survey on Neural Network Interpretability. IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 5, no. 5, pp. 726-742, Oct. 2021.

[5] C. E. Van Wagner. Structure of the Canadian forest fire weather index. Information Canada Department of the Environment, Canadian Forest Service, Publication No. 1333, 1974.

[6] I. Prapas, S. Kondylatos, I. Papoutsis, G. Camps-Valls, M. Ronco, M. – Á. Fernández-Torres, M. P. Guillem, and N. Carvalhais. Deep Learning Methods for Daily Wildfire Danger Forecasting. arXiv:2111.02736 [cs], Nov. 2021.

[7] S. Kondylatos, I. Prapas, M. Ronco, I. Papoutsis, G. Camps-Valls, M. Piles, M. – Á. Fernández-Torres, and N. Carvalhais. Wildfire Danger Prediction and Understanding with Deep Learning. Geophysical Research Letters, vol. 49, no. 17, p. e2022GL099368, Sep. 2022.

[8] S. Kondylatos, I. Prapas, G. Camps-Valls, and I. Papoutsis. Mesogeos: A multi-purpose dataset for data-driven wildfire modeling in the Mediterranean. In Advances in Neural Information Processing Systems, vol. 36, pp. 50661–50676, Curran Associates, Inc, Dec. 2023.

[9] S. Kondylatos, I. Prapas, G. Camps-Valls, and I. Papoutsis. Mesogeos: A multi-purpose dataset for data-driven wildfire modeling in the Mediterranean. Zenodo, 2023; https://zenodo.org/records/8036851 [Accessed 28/03/2025]

[10] S. G. Makridakis, S. C. Wheelwright, and R. J. Hyndman, Forecasting: Methods and Applications. Wiley, 3rd edition, 1998. ISBN: 978-0-471-53233-0.

[11] J. De Stefani, "Towards multivariate multi-step-ahead time series forecasting: A machine learning perspective", Doctoral Dissertation, Université Libre de Bruxelles, 2022; https://dipot.ulb.ac.be/dspace/bitstream/2013/340052/4/De_Stefani_Thesis_Published.pdf [Accessed 28/03/2025]

[12] S. Ben Taieb, G. Bontempi, A. Atiya, and A. Sorjamaa. A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. Expert Systems with Applications, vol. 39, no. 8, pp. 7067-7083, Jun. 2012.

[13] C. Molnar. Interpretable Machine Learning: A Guide for Making Black Box Models Explainable, 2nd edition, 2022; https://christophm.github.io/interpretable-ml-book [Accessed 28/03/2025]

[14] J. H. Friedman. Greedy Function Approximation: A Gradient Boosting Machine. The Annals of Statistics, vol. 29, no. 10, Oct. 2001.

[15] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic Attribution for Deep Networks. International Conference on Machine Learning, Mar. 2017

Nearest Neighbor Search with Locality Sensitive Hashing and Multilayer Perceptron model

Panagiotis E. Drivas

ABSTRACT

The Nearest Neighbor Search problem can be defined as follows: Given a set P of n objects in a metric space (X,D), build a data structure that can answer the following query: Given a query object q, return its nearest neighbor NN, which is the object in the set P that is closest to q. The Nearest Neighbor NN technique is a highly efficient and effective approach used in various applications, including pattern recognition, as well as object and text classification. However, when dealing with high-dimensional objects, the problem becomes significantly more challenging due to increased memory requirements and computational complexity. In this thesis, we represent two-dimensional geometric objects such as polygons, polylines and line segments as vectors in R^d. Given a query object and an appropriate distance function, we aim to identify its nearest neighbor NN using the Locality Sensitive Hashing (LSH) algorithm. While the LSH algorithm works particularly well for some metrics such as the Hamming or Euclidean metric, it doesn't work well for other metrics (e.g., the Fréchet metric). To overcome this problem we introduce a mapping, implemented by a Multilayer Perceptron model - MLP, a type of feedforward neural network. The goal of the MLP model is to transform the input metric space to a Euclidean Space, a space where the LSH algorithm is particularly effective.

ADVISOR

Ioannis Emiris, Professor NKUA

1 INTRODUCTION

1.1 Nearest Neighbor Problem

The Nearest Neighbor Search problem can be defined as follows: Given a set P of n objects in a metric space (X, D) where D is the distance function, build a data structure that can answer the following query: Given a query object q, return its nearest neighbor NN, which is the object in the set P that is closest to q.

$$\mathrm{NN(q)} = \min_{\mathrm{p}\in\mathrm{P}} \mathrm{D(q,p)} \quad (1.1)$$

The Nearest Neighbor (NN) technique is a highly efficient and effective approach used in various applications, including pattern recognition, as well as object and text classification. However, when dealing with high-dimensional objects (i.e., $d > 10$), the problem becomes significantly more challenging due to increased memory requirements and computational complexity. More specifically, their space or time requirements grow exponentially in the dimension [1].

A straightforward solution to this problem involves storing the set P in memory and, given a query q, calculating the distance $\mathrm{D(q,p)}$ for each $\mathrm{p} \in \mathrm{P}$ and selecting the point p with the minimum distance. However, this approach is computationally expensive: calculating all n distances requires at least n operations. Consequently, more efficient methods have been developed to find nearest neighbors without explicitly computing all distances from q [2]. One such method, which we will employ in this thesis, is the Locality Sensitive Hashing (LSH) algorithm.

The work focuses on two-dimensional geometric objects such as polygons, polylines, squares, trajectories, and line segments, all represented as vectors in R^d. Given a query object and an appropriate distance function, we aim to identify its nearest neighbor NN using the Locality Sensitive Hashing LSH algorithm. In addition to the nearest neighbor problem, we also consider the k-nearest neighbors $k-NN$ problem, where $k > 1$. We perform experiments for both problems, and report the results in Chapter 4. This problem is of high importance

in machine learning, computer vision, data analysis applications such as data mining, signal processing, protein sequence analysis and geospatial data.

1.2 LSH Algorithm

As mentioned earlier, the straightforward solution to the NN problem becomes computationally expensive, especially for high-dimensional datasets. To address this issue, hashing techniques have been introduced to organize and store data in containers known as hash buckets. However, traditional hashing methods distribute data uniformly, disregarding the underlying spatial relationships between the input objects.

The Locality Sensitive Hashing LSH algorithm is a variant of conventional hashing algorithms, specifically designed to tackle the NN problem. Unlike conventional hashing schemes, LSH increases the probability that similar input objects are hashed into the same bucket, maximizing the chances of collisions among similar objects. Simultaneously, it seeks to minimize the likelihood that dissimilar objects end up in the same bucket, effectively separating them.

The LSH algorithm can be broken down into the following steps:

- **Data Structure:** A data structure is first constructed by hashing the input objects and then dividing them into buckets, based on their hash values. Each bucket contains objects that are similar to one another.
- **Query:** The LSH takes a query object as input and hashes it using a hash function. It then retrieves the corresponding bucket from the data structure, narrowing down the search space to only the objects within this bucket, significantly reducing the number of objects that need to be examined.
- **NN:** Finally, a search is performed within the selected bucket to find the query's k – NN's within the selected bucket.

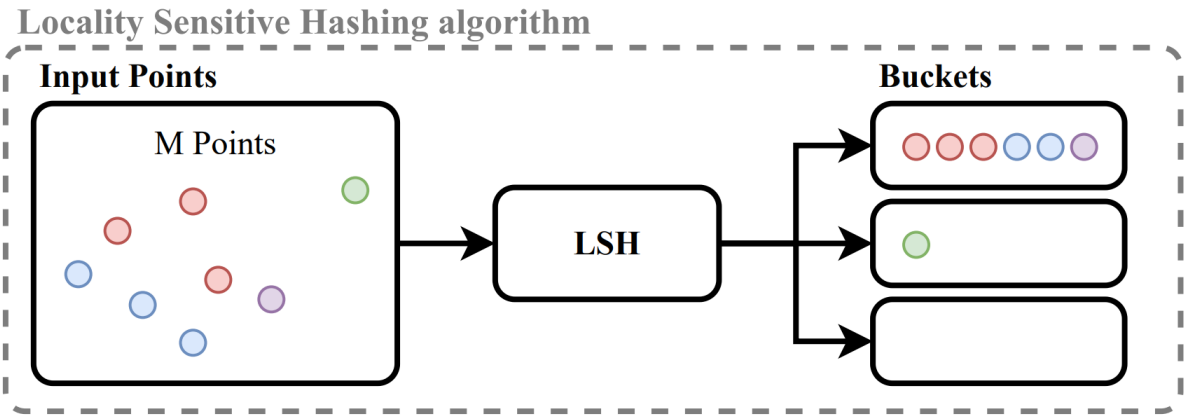


Figure 1: The diagram illustrates how the LSH algorithm works. It hashes the M input points and, after processing them, clusters them into different buckets based on their distances.

1.3 Random Projection

The LSH used in this thesis employs random projection RP, to perform the $k-NN$ search problem. RP involves generating random hyperplanes, each defined by a vector perpendicular to it. These hyperplanes divide the input objects into groups, based on which side of the hyperplane they fall on. To determine the side an object resides on, the dot product between the object (represented as a vector) and the vector defining the hyperplane h is being computed. If the dot product is positive, they reside to the same side and if it is negative they do not. This process is repeated for n randomly chosen and uniformly distributed hyperplanes, generating a lower-dimensional binary vector for each object. The purpose of the RP is to reduce the input d-dimensional objects into lower $k-dimensional$ ones, where $k \ll n$ while preserving the relative distances between objects. RP is not only computational efficient but also a sufficiently accurate method for dimensionality reduction [8].

1.4 k-Nearest Neighbor Graph

The k-Nearest Neighbor Graph $kNNG$ for a set of objects X in a metric space is a directed graph where each object is represented by a vertex in the set V. For each vertex v in V, directed edges are drawn from vertex v to a vertex q whenever q is

one of its $k-NNs$. The NN of each vertex is determined by a specified distance function [9]. To use the $kNNG$ with the set of objects X, a mapping must be established between the vertices V of the graph and the corresponding objects in X.

1.4.1 Soft k-Nearest Neighbor Graph

A variation of the regular $kNNG$, the Soft $kNN\ Graph$ is a weighted graph that allows the connections between the objects in the graph to be more complicated, and be represented by a weight w_i: $0 \leq w_i \leq 1$. Let $\mathrm{X} = \{\mathrm{x1}, \ldots, \mathrm{xN}\}$ be the input dataset, with a metric $\mathrm{d} : \mathrm{X} \times \mathrm{X} \to \mathrm{R} \geq 0$. Given an input hyperparameter k, for each $x_i$we compute the set $\{x_{i1}, \ldots, x_{ik}\}$ of the k-nearest neighbors of x_i under the metric d. Now we can define the Soft $kNN\ Graph$ [10]:

$$\mathrm{G} = (\mathrm{V}, \mathrm{E}, \mathrm{w})$$

, where the vertices V of G are simply the set X, E is the set of directed edges $\mathrm{E} = \{(x_i, x_{i_j}) \mid 1 \leq \mathrm{j} \leq \mathrm{k}, 1 \leq \mathrm{i} \leq \mathrm{N}\}$, and w is the weight function between an object and it's NNs.

1.4.1.1 Adjacency Matrix

The adjacency matrix can be viewed as a low-dimensional representation of the Soft $k-NN-Graph$. Its significance lies in its ability to simplify the complex structure of high-dimensional data while retaining its key patterns and relationships.

The adjacency matrix is a $N \times N$ square matrix (where N is the number of input objects).

It captures the connections between the input objects based on their distances relative to each other.

$$\mathrm{Adj[i,j]} = \begin{cases} w_{ij},\ if\ i \neq j\ \ and\ the\ objects\ \ i\ and\ \ j\ \ are\ connected \\ 0, if\ i = j\ or\ if\ i \neq j\ and\ the\ objects\ i\ and\ j\ are\ not\ connected \end{cases} \quad (1.1)$$

, where w_{ij} is the weight of the connection between input objects i and j, with $0 \leq w_{ij} \leq 1$. A weight w_{ij} with a value closer to 1 indicates a stronger connection

while a value closer to 0 indicates a weaker one. The diagonal elements $\mathrm{Adj}[\mathrm{i},\mathrm{j}] = 0$ indicate no self connections.

1.5 Isometry and Isomorphy

Definition 1.5.1. A function $f : X \rightarrow Y$ between metric spaces (X, d) and (Y, ρ) is called an Isometry if for all $\mathrm{x1}, \mathrm{x2} \in \mathrm{X}$: $\rho(\mathrm{f}(\mathrm{x1}), \mathrm{f}(\mathrm{x2})) = \mathrm{d}(\mathrm{x1}, \mathrm{x2})$.

In this thesis, we approximate the input and output spaces using two distinct Soft $kNN\ Graphs$. The Soft $kNN\ Graphs$ have a special property: given an index of an object in the input space, the corresponding output point will have the same index. We use the Soft $kNN\ Graphs$ and their corresponding adjacency matrices, to preserve the Isomorphy between the two spaces, with the use of the Cross-Entropy loss function between the two Soft $kNNGraphs$.

1.6 Multilayer Perceptron model

While the LSH algorithm works particularly well for some metrics such as the Hamming or Euclidean metric, it doesn't work well for other metrics. To overcome this problem we introduce a mapping, implemented by a Multilayer Perceptron model MLP. The MLP is a type of feedforward artificial neural network. It consists of three or more layers: an input, an output and one or more hidden layers that are not directly connected with the environment. Each layer contains a network of interconnected neurons. The MLP takes as input a dataset of objects (e.g., polylines) and produces a single point of lower size as output for each input object. The goal of the MLP is to transform the input metric space to a Euclidean Space, a space where the LSH algorithm is particularly effective. At the same time, the mapping performed by the MLP attempts to replace the computationally complex Distance Function used in the input space (ex. Fréchet distance) to a simpler and faster one: the $L2\ Norm$.

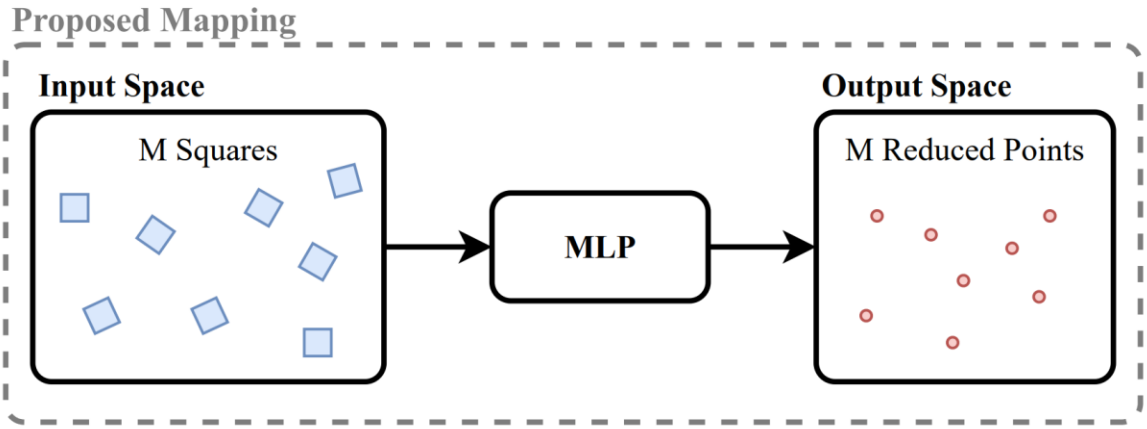


Figure 2: The diagram illustrates the proposed mapping of the M input squares to M reduced points; a process done by the MLP.

1.7 Different learning objectives

For this thesis, an MLP model with two different learning objectives was implemented. The first is the MLP w/Isometry (MLP – [1]) that aims to preserve the Isometry between the two spaces, and the second is the MLP w/Isomorphy (MLP – [2]) that aims to preserve the Isomorphy between the two spaces. In both cases, the MLP tries to preserve some spatial relations of the objects between the input and output spaces. The MLP uses a Loss Function to calculate how far it is from achieving the learning objective. In general, a Loss Function in neural networks is a function $\mathrm{Loss}(\mathrm{y}, \mathrm{x})$ that given an actual result y and a expected result x, calculates the distance between them. The goal of the MLP is to minimize the loss computed by the Loss Function during training.

2 RELATED WORK

2.1 Approximate Nearest Neighbor Problem

The Approximate Nearest Neighbor Problem$((1 + \epsilon, \mathrm{r}) - \mathrm{ANN})$, where $c = 1 + \epsilon$, can be defined as below [14].

Definition 2.1.1. (Approximate Near Neighbor problem) Let $(\mathrm{M}, \mathrm{d}_M)$ be a metric space.

Given $\mathrm{P} \subseteq \mathrm{M}$ and reals $\mathrm{r} > 0$ and $\epsilon > 0$, build a data structure such that for any query $q \in M$, there is an algorithm performing as follows:

- if $\exists \mathrm{p}^* \in \mathrm{P}$ such that $\mathrm{d}_M(\mathrm{p}^*, \mathrm{q}) \leq \mathrm{r}$, then return any point $\mathrm{p}' \in \mathrm{P}$ such that $\mathrm{d}_M(\mathrm{p}', \mathrm{q}) \leq (1+\epsilon) \cdot \mathrm{r}$,
- if $\forall \mathrm{p} \in \mathrm{P}, \mathrm{d}_M(\mathrm{p}, \mathrm{q}) > (1+\epsilon) \cdot \mathrm{r}$, then report "no".

The $((1+\epsilon, \mathrm{r}) - \mathrm{ANN})$, problem was developed as a technique to avoid the curse of dimensionality, a problem that arises as data becomes more complex. [1]. Piotr Indyk [15] introduced a data structure for the $\epsilon - ANN$ problem for the discrete Fréchet metric (X, D). It achieved query time in$\mathrm{O}(\mathrm{m}^{\mathrm{O}(1)} \log \mathrm{n})$, space in $\mathrm{O}(|\mathrm{X}|^{\sqrt{\mathrm{m}}}(\mathrm{m}^{\sqrt{\mathrm{m}}}\mathrm{n})^2)$ and approximation factor $\mathrm{c} = \mathrm{O}(\log \mathrm{m} + \log\log \mathrm{n})$, where m is the maximum length of a sequence and n the maximum number of elements in the data structure.

2.2 Locality Sensitive Hashing algorithm

Piotr Indyk and Rajeev Motwani [1] were the first to introduce the LSH algorithm in 1998. The motivation was to find an algorithm that improves the known bounds of the Nearest Neighbor Problem, and more specifically the approximate Nearest Neighbor Problem $\epsilon - ANN$. For $\epsilon > 0$ and under the l_p norm for $p \in [1, 2]$, requires query time $\mathrm{O}(\mathrm{dn}^{1/(1+\epsilon)})$ and pre-processing $\mathrm{O}(n^{1+\frac{1}{1+\epsilon}} + \mathrm{dn})$.

3 RESULTS

The results from all experiments are presented below. Both the MLP–[1] and the MLP –[2] were tested on every dataset. Each time, the hyper-parameters were tuned to achieve the highest possible accuracy.

3.1 Isometry

The MLP – [1] tries to preserve the Isometry, previously defined, between the input space and the reduced output space. The graph below shows the $top-1$ and $top-2$ accuracy, for both the test and training sets and for different datasets of approximately 1,500 objects.

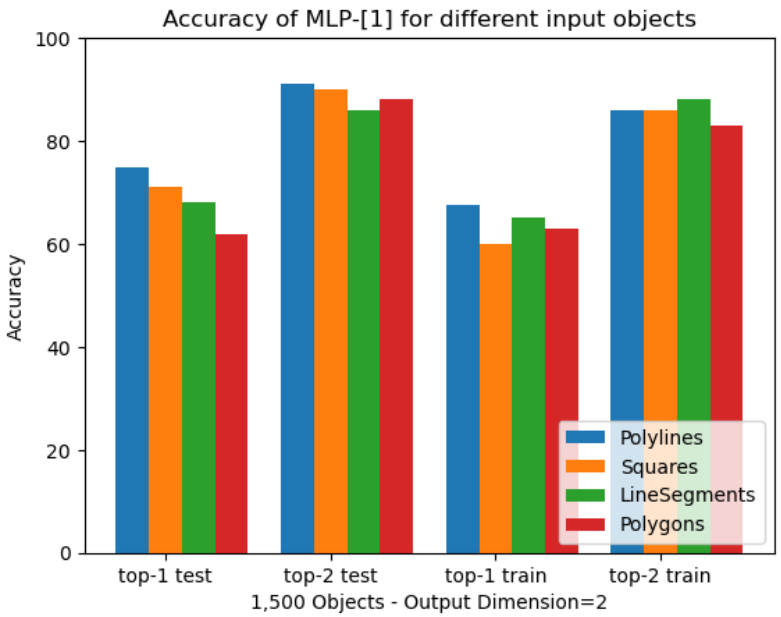


Figure 3: Accuracy of MLP w/Isometry Graph

The $top-1$ and $top-2$ accuracy of the test set is higher than the $top-1$ and $top-2$ accuracy of the training set due to the lower density of the test set. As the complexity of the object in a dataset increases, the accuracy naturally decreases.

3.2 Isomorphy

The MLP – [2] tries to preserve the Isomorphy, previously defined in Chapter 1, between the input space and the reduced output space. In other words, it tries to preserve the relative distances of the input objects from the input space into the output one.

The MLP–[2]'s learning objective is easier to preserve compared to the learning objective of the MLP –[1]. For that reason, it is possible to train the model with larger input datasets in a relatively short amount of time.

3.2.1 Trajectories

The accuracy of MLP – [2] for the dataset of 1,500 Trajectories can be seen in the figure 4.

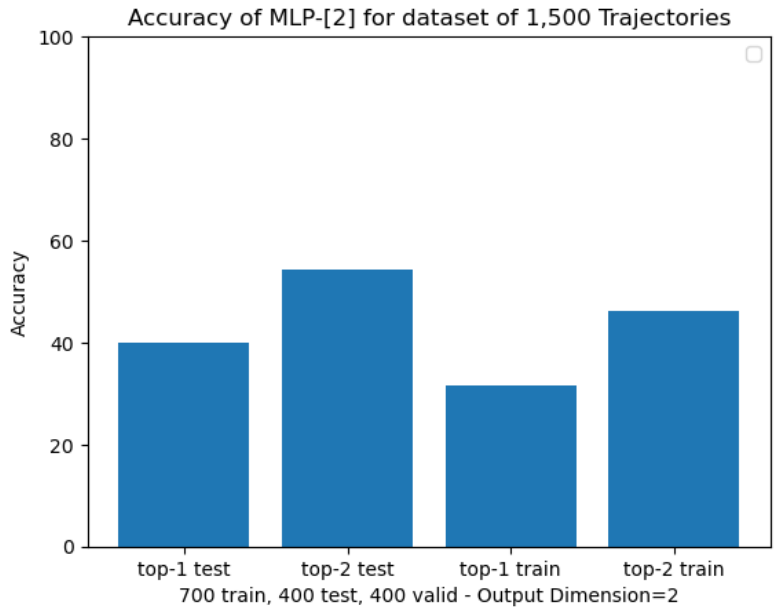


Figure 4: Accuracy of MLP w/Isomorphy for Trajectory Dataset Graph

The MLP – [2] has much higher top – 1 and top – 2 accuracy compared to the MLP – [1]. On one hand, the MLP – [1] collapsed and was not able to retain much information and on the other hand, the MLP – [2] was able to preserve the Isomorphy between the input and output space.

Specifically for the test set, the MLP – [2] increased the $top - 1$ accuracy by +120% and increased the $top - 2$ accuracy by +127%. For the training set, the MLP –[2] increased the $top - 1$ accuracy by +117% and increased the $top - 2$ accuracy by +135% compared the M LP–[1].

3.3 Test Set

Table 4 shows the $top - 1$ and $top - 2$ accuracy of both the MLP's

	Isometry		**Isomorphy**	
	Top-1	**Top-2**	**Top-1**	**Top-2**
Line Segments	68	86	70	91
Squares	71	90	78.8	94.5
Polylines	75	91	72	87.5
Polygons	62	88	72.25	89.9
Trajectories	10	12	40	54.25

Table 4: Accuracy Comparison between Isometry and Isomorphy Models for the test set.

4 CONCLUSIONS

4.1 Increase the Output Point's Dimension of MLP – [2] (Trajectories)

The distance functions used in the experiments are not easily preserved with an output point in R^2. Therefore, the dimension n of the output point was increased to better capture the spatial information of the input space.

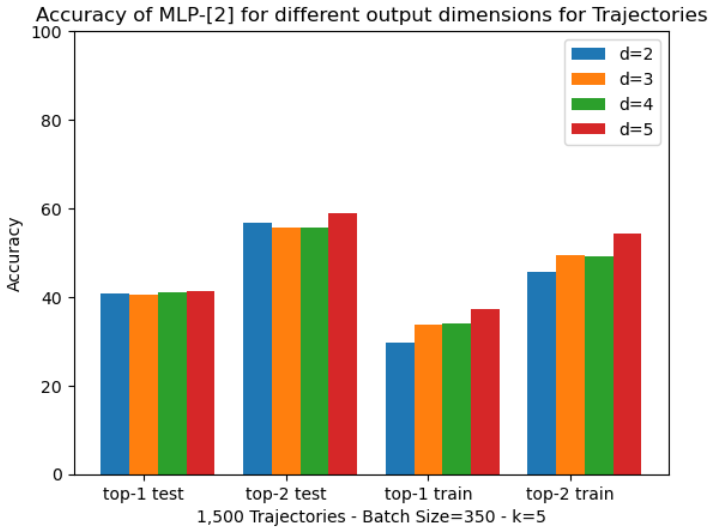


Figure 5: Output Dimension for Trajectory Dataset Graph

4.2 Batch Size of MLP – [2]

Using a large batch size, close to half the training set size, maintained high $top-1$ and $top-2$ accuracy of the MLP–[2]. In contrast, smaller batch sizes with a size close to that of 9%– 18% of the dataset resulted in a decrease of the accuracy of the MLP–[2].

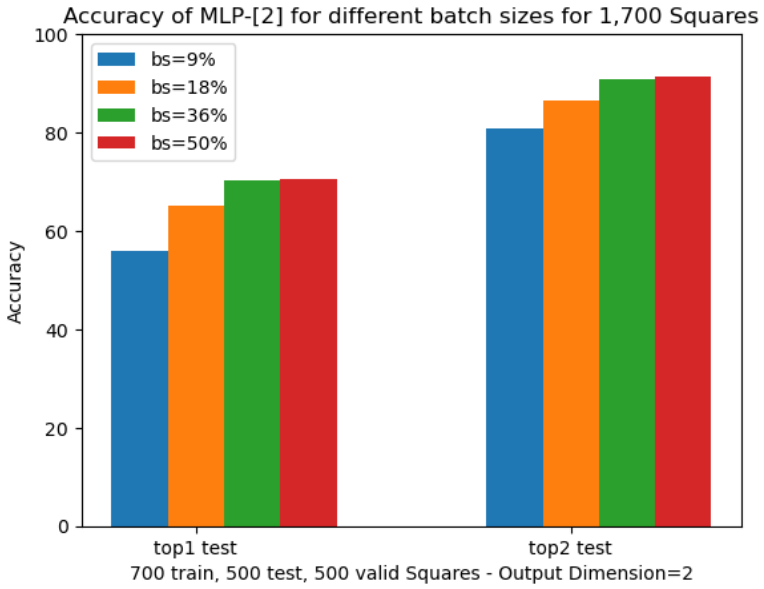


Figure 6: Batch Size for the MLP w/Isomorphy Graph

REFERENCES

[1] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, STOC '98, page 604–613, New York, NY, USA, 1998. Association for Computing Machinery.

[2] Alexandr Andoni, Piotr Indyk, and Ilya Razenshteyn. Approximate nearest neighbor search in high dimensions, 2018.

[3] Wikipedia contributors. Norm (mathematics) — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Norm_(mathematics)&oldid=1240703189, 2024. [Online; accessed 17-August-2024].

[4] Anne Driemel, Ivor van der Hoog, and Eva Rotenberg. On the discrete fréchet distance in a graph, 2022.

[5] Thomas Eiter and Heikki Mannila. Computing discrete frechet distance. 05 1994.

[6] Omid Jafari, Preeti Maurya, Parth Nagarkar, Khandker Mushfiqul Islam, and Chidambaram Crushev. A survey on locality sensitive hashing algorithms and their applications, 2021.

[7] Wikipedia contributors. Dot product — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Dot_product&oldid=1237145072, 2024. [Online; accessed 6-October-2024].

[8] Ella Bingham and Heikki Mannila. Random projection in dimensionality reduction: applications to image and text data. In Knowledge Discovery and Data Mining, 2001.

[9] Wei Dong, Moses Charikar, and K. Li. Efficient k-nearest neighbor graph construction for generic similarity measures. In The Web Conference, 2011.

[10] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020.

[11] Wikipedia contributors. Fuzzy set — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Fuzzy_set&oldid=1243184184, 2024. [Online; accessed 12-September-2024].

[12] Marius-Constantin Popescu, Valentina Balas, Liliana Perescu-Popescu, and Nikos Mastorakis. Multilayer perceptron and neural networks. WSEAS Transactions on Circuits and Systems, 8, 07 2009.

[13] Wikipedia contributors. Activation function — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Activation_function&oldid=1244898906, 2024. [Online; accessed 11-September-2024].

[14] Georgia Avarikioti, Ioannis Z. Emiris, Ioannis Psarros, and Georgios Samaras. Practical linear-space approximate near neighbors in high dimension. CoRR, abs/1612.07405, 2016.

[15] Piotr Indyk. Approximate nearest neighbor algorithms for frechet distance via product metrics. In Proceedings of the Eighteenth Annual Symposium on Computational Geometry, SCG '02, page 102–106, New York, NY, USA, 2002. Association for Computing Machinery.

[16] Alexandr Andoni and Ilya P. Razenshteyn. Optimal data-dependent hashing for approximate near neighbors. CoRR, abs/1501.01062, 2015.

Study and Survey of Nonlinear Dimensionality Reduction Methods in Multidimensional Data

Georgios J. Zorpidis, Konstantinos M. Fragkos

ABSTRACT

This thesis aims to study and review the most widely used nonlinear dimensionality reduction techniques and methods for multidimensional data. It focuses on examining the mathematical principles that underpin these methods while also presenting experimental results on different datasets and techniques for improving them using pre¬trained models. The thesis compares these methods based on various metrics, highlighting their limitations and drawing conclusions about their effectiveness.

Subject Area: Dimensionality Reduction

Keywords: dimensionality reduction, nonlinear dimensionality reduction methods, deep learning and dimensionality reduction, data visualization, machine learning, manifolds

ADVISOR

Ioannis Emiris, Professor NKUA

1 INTRODUCTION

Dimensionality reduction is a crucial aspect of machine learning and data analysis, enabling the simplification of high-dimensional datasets while preserving their essential structure. As data grows in complexity, the need for

efficient visualization, analysis, and processing methods becomes increasingly important. This thesis investigates the fundamental problem of dimensionality reduction, focusing on various techniques that aim to transform high-dimensional data into more manageable representations while retaining meaningful information. To define the problem of dimensionality reduction, we first have to assume a dataset as a matrix $X \in \mathbb{R}^{n\,X\,D}$, where we have n samples of dimensionality D each. Our goal is to find a mapping $f : R^{D} \rightarrow R^{d}$, where $d << D$, so that the data is represented in a lower dimensional space. This mapping should preserve the geometry of the data as well as possible. We also assume that the points in the dataset X lie on, or near a manifold [19] in the high dimensional space. Since in general, the geometry of the datasets is not known beforehand the problem of findings such a mapping, or representation, is a challenging one.

Numerous studies have explored dimensionality reduction techniques, establishing a strong foundation in both theoretical and applied domains. Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are among the most widely used linear methods, offering efficient ways to reduce dimensionality while maximizing variance and class separability, respectively. Nonlinear techniques such as Isomap, Locally Linear Embedding (LLE), Hessian LLE, t-Distributed Stochastic Neighbor Embedding (t-SNE), and Uniform Manifold Approximation and Projection (UMAP) have gained significant attention for their ability to preserve complex manifold structures in high-dimensional spaces. Recent advancements in deep learning have also introduced autoencoders and contrastive learning-based approaches, further expanding the capabilities of dimensionality reduction.

This thesis builds upon these prior studies by systematically reviewing and comparing these methods, highlighting their strengths and limitations. It aims to examine the theoretical underpinnings of dimensionality reduction methods, distinguishing between linear and nonlinear approaches and analyze experimental results obtained from applying these techniques to benchmark datasets using benchmarking metrics that remain underdeveloped in the current literature and have not been extensively developed, as well as their suitability for tasks such as clustering, classification, and visualization. We also

Investigate the integration of dimensionality reduction with pretrained machine learning models to enhance performance and interpretability.

2 SURVEY OF DIMENSIONALITY REDUCTION METHODS

2.1 Linear Methods

Linear methods such as Principal Component Analysis [22], Linear Discriminant Analysis [7] and Canonical Correlation Analysis [8] all include basic techniques are foundational in data analysis but exhibit notable limitations when applied to complex, real world datasets. For example, linear techniques are highly sensitive to noise and outliers, and they capture only the global structure of the data while ignoring local patterns. Furthermore, these methods assume that the data distribution remains consistent across all samples, an assumption that often doesn't hold true in real-world datasets. These limitations of linear techniques and some more like these have been researched and documented thoroughly and extensively [9].

2.2 Non-linear Manifold Learning Techniques

The non-linear techniques that will be explored below fall under the umbrella of manifold learning, which operates on the premise that high dimensional data often resides on low dimensional manifolds embedded within the higher dimensional space.

Laplacian Eigenmaps. The Laplacian Eigenmap [2] method requires a deep mathematical understanding. Initially, the algorithm begins with the creation of two matrices based on the data and their neighborhood relationships, namely Adjacency and Degree matrices, based on the data and their neighborhood relationship and lastly creating the Laplacian matrix, for which we aim to select the smallest eigenvalues, which show the directions of the new basis vectors with the least variance, where the data is most clustered.

Local Linear Embedding. The Local Linear Embedding (LLE) [16] technique is a method with many similarities to Laplacian Eigenmaps. The technique creates a graph to represent the data and focuses on the local neighborhoods of the data.

Isomap. The Isometric Mapping method (ISOMAP) [18] is based on some basic principles of MDS, a linear method that is extended to work on Manifolds without the linear constraint, so as to better generalize complex structures. This is achieved with the basic metric of geodesic distance.

UMAP. The Uniform Manifold Approximation and Projection (UMAP) [13] is one of the most recent techniques for dimensionality reduction. UMAP is centered around two main steps: 1. Creation of topological data representation 2. Finding the optimal representation in the latent space

T-SNE. The t-distributed Stochastic Neighbor Embedding (tSNE) [19] method is a dimensionality reduction method developed by Laurens van der Maaten and Geoffrey Hinton. It is considered one of the most widespread and effective dimensionality reduction algorithms. It is used extensively to map data to lower dimensions (usually 2 or 3).

2.3 Deep learning methods

Parametric UMAP and T-SNE. Combinations of the algorithms (UMAP and T-SNE) with a neural network.

Deep Autoencoder. A deep autoencoder is a type of Artificial neural network which learns data encodings in an unsupervised way. The aim of the deep autoencoder is to learn the lower data representations of the higher data representations by training the network and keeping the most important features and characteristics.

Convolutional Autoencoder. Convolutional autoencoder is a type of autoencoder which tries to reduce the noise from data by learning the underlying structure of the input. [15] The structure consists of an encoder and a decoder.

Variational Autoencoder. Assuming a set of datapoints came from an unknown distribution $p(x)$, the main purpose of variational autoencoders is to map the

datapoints from the unknown $p(x)$ to a known prior distribution $p(z)$. This is going to happen by using the posterior distribution $p(z|x)$ for every datapoint x to make the mapping from x space to z space and the likelihood $p(x|z)$ to make the opposite mapping.

2.4 Pretrained models

While experimenting with the above methods, we came across various hardships, especially with difficult datasets. To combat this, we decided to add the intermediate step of using pretrained models to bring the initial dimensionality lower and then use the methods as usual. Since all the datasets we used were image based datasets, we decided to use pretrained models that were trained on images. Specifically, we used 2 pretrained models, the VGG16 model, and the RESNET model. VGG16 is a convolutional neural network with 16 layers, trained to classify 1000 categories, while ResNet50 is a 50layer residual network. The effect of their usage is discussed in the insights part of the next chapter.

3 EXPERIMENTAL COMPARISON AND DISCUSSION

3.1 Experimental Setup

The experimental setup consists of the main metrics and datasets used. For the evaluation metrics, we firstly took into account the accuracy of a KNN classifier trained on the embedded data. Then we have local metrics, emphasizing on local attributes, that include Trustworthiness and Continuity. The first one, measures how many of the nearest neighbours in the original space remain nearest neighbours in the embedded space and the second one the inverse.

Global metrics that emphasize the global structure of the data, namely Stress and the Kullback-Leibler divergence. Stress, is designed to evaluate how well the embedding preserves the original distances among data points and the KL divergence measures the difference between the distribution of data points in the original space with the distribution in the latent space.

Lastly, we have clustering metrics: Silhouette Score, Distance Consistency, Cohesiveness, Steadiness. The silhouette score evaluates how well data points are clustered by measuring the cohesion within clusters and the separation between clusters. Distance Consistency measures the proportion of points in an embedding that are closer to their corresponding class centroid than to the centroids of other classes. Cohesiveness, measures the tightness of clusters or groups of points in the embedded space compared to their counterparts in the original space and Steadiness measures how consistently the local structure of the data is preserved in the embedding.

As far as the datasets are concerned, we have both artificial and real-world datasets, from the artificial that include 3 planes, 2 spiral, Dollar sign datasets to PneumoniaMNIST, MNIST, CIFAR-10 and SmallNorb datasets.

3.2 Results and Analysis

Table 1. PneumoniaMNIST dataset results

Method	D	Trust	Cont	Stead	Coh.	Sil.	Stress	KL	D.C.	KNN
t-SNE	2	0.942	0.928	0.783	0.726	0.257	10.132	0.015	0.877	0.941
UMAP	2	0.905	0.940	0.730	0.748	0.275	0.335	0.013	0.880	0.936
Isomap	2	0.842	0.947	0.655	0.710	0.278	0.559	0.010	0.878	0.895
L.E.	2	0.977	0.991	0.875	0.820	0.082	0.999	0.078	0.923	0.910
LLE	2	0.851	0.936	0.677	0.716	0.207	0.995	0.010	0.841	0.840
D.A.	2	0.888	0.894	0.665	0.743	0.352	0.867	0.014	0.841	0.848
C.A.	2	0.839	0.919	0.628	0.734	0.211	1.080	0.015	0.782	0.846
V.A.	2	0.891	0.918	0.662	0.748	0.353	0.939	0.011	0.880	0.860
t-SNE	3	0.958	0.943	0.821	0.751	0.198	3.685	0.015	0.845	0.949
UMAP	3	0.935	0.961	0.787	0.732	0.250	0.370	0.011	0.875	0.928
Isomap	3	0.912	0.973	0.775	0.760	0.235	0.613	0.008	0.883	0.901
L.E.	3	0.977	0.991	0.878	0.830	0.082	0.999	0.078	0.923	0.910
LLE	3	0.876	0.935	0.698	0.721	0.232	0.994	0.019	0.841	0.877
D.A.	3	0.922	0.911	0.745	0.749	0.255	0.625	0.011	0.865	0.869
C.A.	3	0.885	0.949	0.725	0.754	0.152	0.586	0.019	0.824	0.851
V.A.	3	0.881	0.929	0.656	0.749	0.303	0.933	0.016	0.809	0.843

Table 2. PneumoniaMNIST dataset results using custom Nueral Network

Method	D	Trust	Cont	Stead	Coh.	Sil.	Stress	KL	D.C.	KNN
t-SNE	2	0.946	0.921	0.736	0.833	0.289	1.730	0.030	0.893	0.934
UMAP	2	0.893	0.934	0.681	0.823	0.320	0.828	0.029	0.908	0.926
Isomap	2	0.816	0.942	0.615	0.793	0.294	0.478	0.023	0.905	0.922
L.E.	2	0.829	0.922	0.615	0.809	0.437	1.000	0.033	0.923	0.929
LLE	2	0.759	0.865	0.522	0.751	0.196	0.999	0.024	0.844	0.834
t-SNE	3	0.959	0.929	0.787	0.815	0.236	0.425	0.027	0.898	0.938
UMAP	3	0.931	0.951	0.757	0.828	0.322	0.847	0.026	0.920	0.927
Isomap	3	0.888	0.968	0.743	0.817	0.243	0.568	0.018	0.910	0.910
L.E.	3	0.984	0.993	0.881	0.870	0.072	1.000	0.049	0.923	0.926
LLE	3	0.796	0.867	0.630	0.766	0.233	0.999	0.024	0.868	0.870

Table 3. MNIST dataset results

Method	D	Trust	Cont	Stead	Coh.	Sil.	Stress	KL	D.C.	KNN
t-SNE	2	0.975	0.964	0.802	0.901	0.351	6.849	0.140	0.880	0.948
UMAP	2	0.960	0.964	0.785	0.932	0.438	0.444	0.161	0.906	0.946
Isomap	2	0.766	0.948	0.489	0.715	-0.015	0.779	0.101	0.454	0.688
L.E.	2	0.964	0.985	0.826	0.856	0.102	1.000	0.087	0.628	0.841
LLE	2	0.747	0.910	0.477	0.783	-0.044	0.999	0.299	0.414	0.713
D.A.	2	0.924	0.907	0.666	0.846	-0.007	0.695	0.168	0.535	0.719
C.A.	2	0.821	0.914	0.558	0.864	0.030	0.568	0.296	0.539	0.493
V.A.	2	0.900	0.918	0.656	0.867	0.087	0.888	0.189	0.648	0.641
t-SNE	3	0.985	0.971	0.838	0.921	0.297	1.721	0.139	0.895	0.946
UMAP	3	0.976	0.968	0.813	0.928	0.413	0.442	0.144	0.902	0.945
Isomap	3	0.865	0.970	0.695	0.802	0.047	2.020	0.081	0.548	0.746
L.E.	3	0.987	0.992	0.862	0.887	0.112	1.000	0.070	0.764	0.883
LLE	3	0.882	0.932	0.632	0.873	0.111	0.998	0.336	0.528	0.855
D.A.	3	0.954	0.943	0.764	0.876	0.080	0.447	0.204	0.623	0.768
C.A.	3	0.900	0.955	0.707	0.834	0.078	0.408	0.143	0.618	0.636
V.A.	3	0.949	0.948	0.769	0.887	0.117	0.884	0.180	0.746	0.738

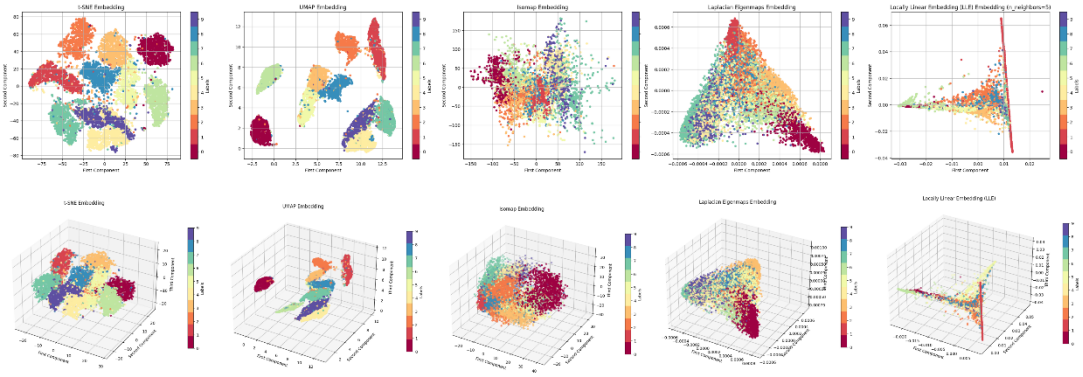


Fig. 1: MNIST plots of the main Manifold learning algorithms for dimensions 2 and 3

Table 4. CIFAR10 dataset results for default

Method	D	Trust	Cont	Stead	Coh.	Sil.	Stress	KL	D.C.	KNN
t-SNE	2	0.900	0.905	0.793	0.520	-0.137	0.456	0.028	0.198	0.292
UMAP	2	0.836	0.932	0.709	0.528	-0.138	0.906	0.028	0.201	0.184
Isomap	2	0.821	0.938	0.684	0.523	-0.123	0.413	0.020	0.201	0.177
L.E.	2	0.931	0.983	0.859	0.595	-0.037	1.000	0.063	0.281	0.277
LLE	2	0.808	0.917	0.676	0.509	-0.136	1.000	0.027	0.200	0.172
D.A.	2	0.828	0.810	0.677	0.471	-0.279	12.22	0.046	0.183	0.176
C.A.	2	0.797	0.887	0.663	0.495	-0.172	0.875	0.040	0.179	0.146
V.A.	2	0.855	0.890	0.737	0.501	-0.114	0.930	0.032	0.215	0.166
t-SNE	3	0.900	0.905	0.793	0.520	-0.137	0.456	0.028	0.198	0.285
UMAP	3	0.883	0.951	0.795	0.499	-0.126	0.915	0.026	0.219	0.221
Isomap	3	0.874	0.959	0.772	0.528	-0.104	0.383	0.017	0.223	0.201
L.E.	3	0.821	0.931	0.676	0.533	-0.111	1.000	0.050	0.209	0.173
LLE	3	0.826	0.921	0.705	0.514	-0.110	0.999	0.028	0.211	0.195
D.A.	3	0.858	0.853	0.739	0.453	-0.218	1.813	0.048	0.196	0.203
C.A.	3	0.842	0.922	0.733	0.462	-0.145	0.787	0.029	0.218	0.175
V.A.	3	0.895	0.922	0.806	0.505	-0.096	0.937	0.027	0.240	0.194

Table 5. CIFAR10 dataset results using RESNET pretrained model

Method	D	Trust	Cont	Stead	Coh.	Sil.	Stress	KL	D.C.	KNN
t-SNE	2	0.942	0.934	0.756	0.855	0.179	1.799	0.075	0.728	0.831
UMAP	2	0.910	0.941	0.711	0.880	0.191	0.804	0.074	0.721	0.807
Isomap	2	0.777	0.920	0.565	0.791	-0.042	0.708	0.064	0.472	0.445
L.E.	2	0.812	0.919	0.599	0.802	-0.012	1.000	0.095	0.522	0.536
LLE	2	0.810	0.902	0.607	0.824	-0.033	0.999	0.109	0.531	0.600
t-SNE	3	0.954	0.943	0.812	0.822	0.157	0.495	0.056	0.771	0.833
UMAP	3	0.933	0.948	0.769	0.872	0.214	0.822	0.075	0.792	0.828
Isomap	3	0.867	0.956	0.742	0.782	0.039	0.873	0.053	0.630	0.630
L.E.	3	0.965	0.984	0.866	0.828	0.138	1.000	0.050	0.780	0.790
LLE	3	0.855	0.921	0.677	0.820	0.009	0.999	0.100	0.534	0.675

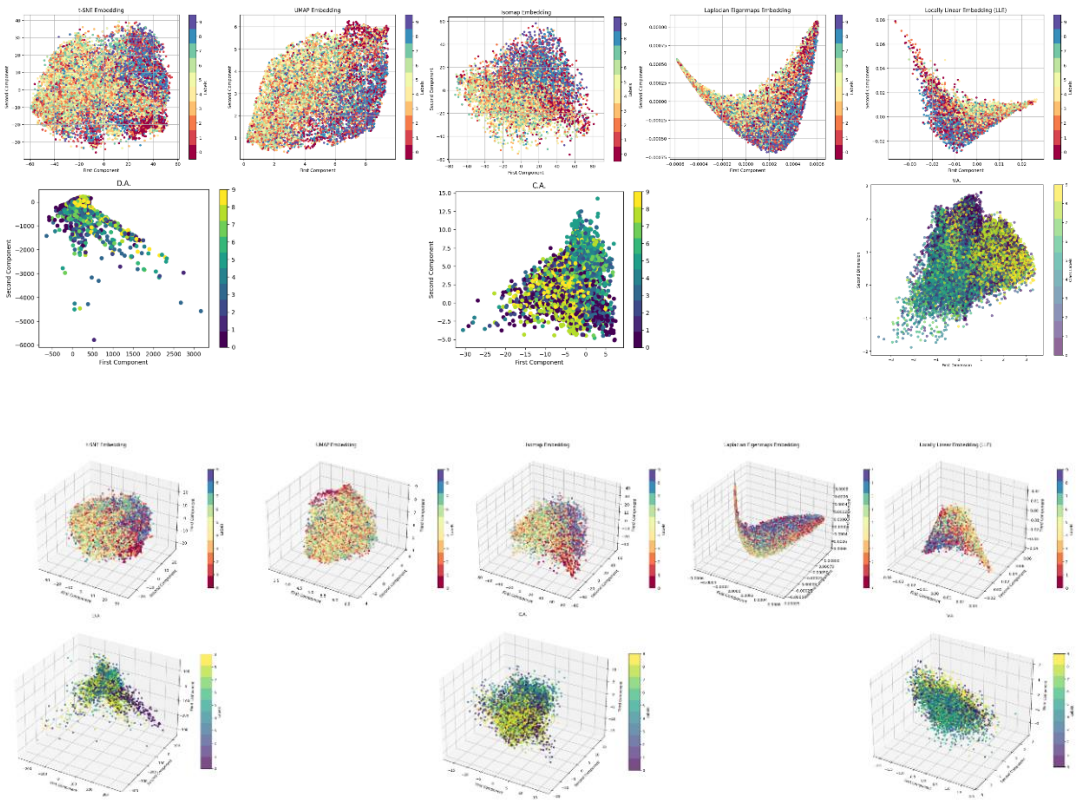


Fig. 2: Plots of methods in 2 and 3 dimensions for default CIFAR10

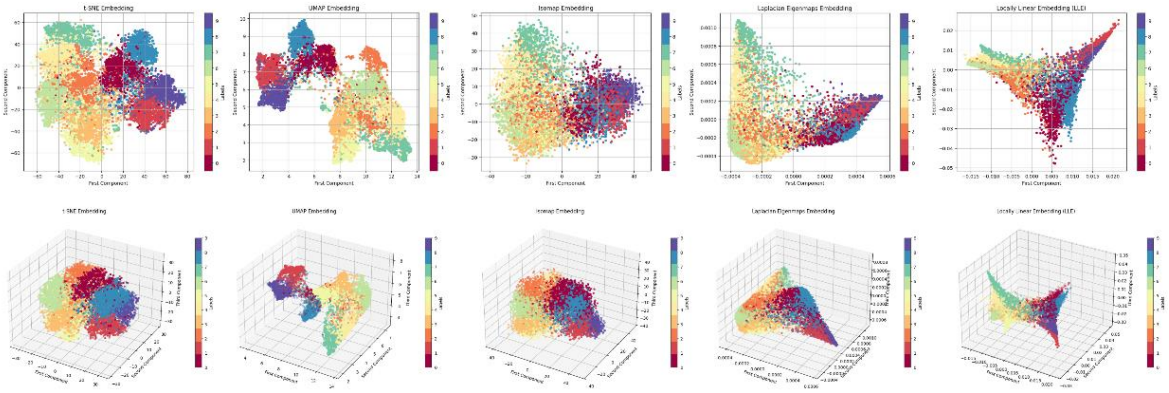


Fig. 3: Plots of methods in 2 and 3 dimensions for CIFAR10 using RESNET50

Table 6. smallNorb dataset results for default

Method	Dim	Trust	Cont	Stead	Coh.	Sil.	Stress	KL	D.C.	KNN
t-SNE	2	0.983	0.977	0.831	0.781	-0.022	3.195	0.040	0.395	0.948
UMAP	2	0.978	0.978	0.819	0.793	-0.044	0.520	0.024	0.285	0.863
Isomap	2	0.906	0.965	0.558	0.706	-0.045	0.759	0.012	0.263	0.463
L.E.	2	0.974	0.980	0.819	0.770	-0.031	1.000	0.149	0.478	0.719
LLE	2	0.774	0.884	0.295	0.603	-0.089	0.999	0.054	0.283	0.383
D.A.	2	0.940	0.945	0.628	0.703	-0.062	0.965	0.015	0.258	0.552
C.A.	2	0.925	0.966	0.593	0.686	-0.026	0.857	0.027	0.272	0.466
V.A.	2	0.697	0.887	0.368	0.610	-0.084	0.999	0.044	0.356	0.463
t-SNE	3	0.986	0.979	0.870	0.830	-0.015	0.804	0.047	0.436	0.966
UMAP	3	0.984	0.982	0.847	0.767	-0.022	0.543	0.026	0.339	0.896
Isomap	3	0.938	0.979	0.744	0.684	-0.024	0.820	0.008	0.343	0.640
L.E.	3	0.974	0.980	0.812	0.776	-0.031	1.000	0.149	0.478	0.719
LLE	3	0.836	0.919	0.450	0.646	-0.055	0.999	0.107	0.405	0.573
D.A.	3	0.963	0.958	0.693	0.689	-0.063	0.942	0.064	0.354	0.723
C.A.	3	0.964	0.983	0.772	0.752	-0.036	0.783	0.058	0.317	0.640
V.A.	3	0.898	0.854	0.492	0.631	-0.124	0.780	0.031	0.300	0.449

Table 7. smallNorb dataset results using VGG16 pretrained model

Method	D	Trust	Cont	Stead	Coh.	Sil.	Stress	KL	D.C.	KNN
t-SNE	2	0.989	0.983	0.849	0.733	0.306	0.461	0.162	0.785	0.999
UMAP	2	0.984	0.985	0.829	0.730	0.454	0.842	0.150	0.877	1.000
Isomap	2	0.793	0.943	0.427	0.647	0.032	1.271	0.143	0.590	0.701
L.E.	2	0.974	0.989	0.777	0.798	0.122	1.000	0.100	0.939	0.983
LLE	2	0.832	0.954	0.385	0.662	0.209	1.000	0.151	0.685	0.934
t-SNE	3	0.993	0.987	0.879	0.733	0.253	0.621	0.151	0.892	1.000
UMAP	3	0.988	0.988	0.833	0.737	0.455	0.853	0.172	0.898	1.000
Isomap	3	0.881	0.966	0.563	0.677	0.107	1.388	0.112	0.667	0.846
L.E.	3	0.974	0.989	0.763	0.793	0.122	1.000	0.100	0.939	0.983
LLE	3	0.855	0.959	0.481	0.711	0.174	1.000	0.152	0.712	0.943

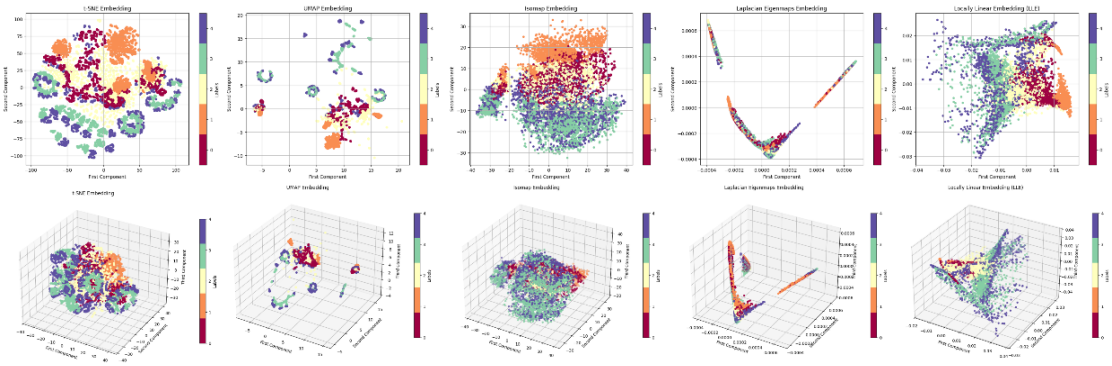


Fig. 4: Plots of methods in 2 and 3 dimensions for default smallNorb

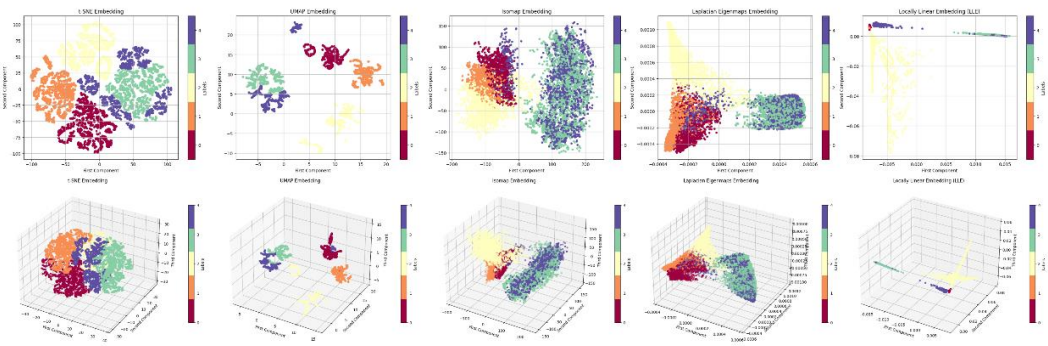


Fig. 5: Plots of methods in 2 and 3 dimensions for CIFAR10 using RESNET50

3.3 Insights and Implications

Trends and Patterns Observed in the Results. Based on the above experiments, we concluded to some interesting results. We will begin the discussion with the more general observations and then move to the more specific ones. Firstly, while trustworthiness and KNN accuracy can seem interconnected, in reality and as shown by our results, this is not the case. A high Trustworthiness score does not imply a high KNN accuracy score. If the translation of these neighbors from the high dimensional space to the low dimensional space is good, yielding a high Trustworthiness score, it may be the case that these neighbors do not have the same label, in neither space. Thus, the KNN accuracy score will be low.

Moving on, Steadiness, which measures how consistently the local structure of the data is preserved in the embedding, is relatively medium and above. The only exceptions seem to be Isomap and LLE, which for two different reasons, seem to underperform. Isomap is designed to preserve global structure by approximating the geodesic distance between points in the highdimensional space. While this approach works well for capturing the overall geometry of the data, it may not always retain local relationships between points as effectively. LLE on the other hand, prioritizes local relationships in the data, thus hindering its ability to capture the global structure of the data. This is reflected in the relatively low Steadiness scores for both methods.

Thirdly, Continuity and Cohesiveness, which both attent to the global structure of the data, seems to be the relatively the most consistent metrics across all methods and datasets, being relatively high in most experiments, excluding the very ill performing CIFAR10 dataset which yields bad results for all metrics.

Fourthly, the Silhouette scores, which measures how similar an object is to its own cluster compared to other clusters, seem to be hovering around the median value of 0, meaning that most methods do an average job at creating clusters. However, tSNE and UMAP seem to be the best performers in this regard, with the highest scores across all experiments. This is expected, as both methods are designed to create clusters in the low-dimensional space. Another important correlation is that high Silhouette scores seem be correlated with high KNN accuracy score. This is expected, since when data is clustered well, the KNN

classifier will perform better, picking from surrounding points that are similar to the query point. On the other hand, when Silhouette scores are relatively low (near 0), the KNN accuracy score is also low, as the data is not clustered well, and the KNN classifier will have a hard time finding similar points to the query point due to fuzzy bounderies.

Fifthly, the Stress metric, which measures how well the distances between points in the highdimensional space are preserved in the low-dimensional space, also seem rather consistent across all experiments. Isomap, which inherently minimizes the Stress function is not always the best performing method for reasons explained above.

The KL divergence seems to be performing rather well across all experiments, leaning relatively close to 0, with some expections, mainly on LLE and Convolutional Autoencoders, due to these methods not prioritizing learning the underlying probability distributions of the data. Other methods, like Laplacian Eigenmaps, tSNE or UMAP seem to be performing excellent. Laplacian Eigenmaps do not inherently minimize the KL divergence however. The method constructs a graph where nodes represent data points, and edges represent similarities between them. The key idea is to preserve the global structure of the data by minimizing a smoothness criterion that ensures that connected points remain close in the low-dimensional embedding. The optimization of the Laplacian eigenvectors naturally preserves global relationships between data points, which means that even if local distances are emphasized, the overall structure is maintained in a way that results in good global distance approximations. KL Divergence and Global Structure: Since KL divergence measures the difference in probability distributions over pairwise distances, methods that preserve global structures (like Laplacian Eigenmaps) are more likely to maintain the overall pairwise relationships, leading to a low KL divergence. In other words, the smoothness enforced by the Laplacian eigenmaps minimization process means that the pairwise distances between points in the original high-dimensional space are well approximated in the lower dimensional embedding.

Moreover, the Distance consistency metric, which differentiates itself by using the labels of the data, seems to be fluctuating a lot. This is expected, due to the way the metric is measured, by creating artificial centroids and measuring the distance between the centroids and the data points. It can be seen that when the Distance consistency is high, meaning $\gtrapprox 0.8$, KNN accuracy scores are also high. This is intuitively understood, since if Distance consistency is high, that means that the computed centroids are close to the data points, and the KNN classifier will have an easier time finding the correct neighbors. However, the inverse is not necessarily true. Since centroids can be computed and be placed on top of other data points, having different labels, which can give high KNN scores, due to potential clusters, but low Distance consistency scores, due to the centroids placement.

Recommendations for Practitioners and Future Research. It recommended to use tSNE and UMAP for most of the cases, in order to visualize datasets in the 2 and 3 dimensional spaces and get a better understanding of the data and complexity of it, based on the clustering quality. It is also recommended to use most of the methods in combination with pretrained models, or using their parametric versions, as they will be able to improve the results significantly. Lastly, it is recommended to use the appropriate metric optimization for appropriate task. For example, for clustering we want to maximize the silhouette score, and UMAP seems to perform better than other methods on that aspect. For classification, we want to maximize the KNN score, and tSNE seems to perform better than other methods on that aspect. For future research, it is recommended to explore the use of more complex neural networks, with more layers, in order to improve the results of the methods. Lastly, it is recommended to explore the use of more complex methods, like the fusion of linear and non-linear methods, and further optimization of all the methods' hyperparameters as well as optimizing the KNN classifier, using cross validation and grid search.

4 CONCLUSION

The survey and experiments provide a comprehensive overview of nonlinear dimensionality reduction methods, highlighting their strengths and weaknesses. The survey covers manifold learning, autoencoders, and other neural networks, while the experiments evaluate the performance of these methods on various datasets. Key insights include the importance of preserving local and global structures, the impact of dataset complexity on method performance, and the potential for improvement through hyperparameter optimization and model selection. The survey and experiments reveal that no single method is universally superior, with each having unique advantages and limitations. The results underscore the need for practitioners to carefully select methods based on dataset characteristics and research goals.

REFERENCES

[1] D. Barman, A. Hasnat, and R. Nag, "An Introduction to Autoencoders" in Computation, 3rd ed., CSE-GCETTB, Feb. 2022, pp. 14–23.

[2] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction," Neural Computation, vol. 15, no. 6, pp. 1373–1396, Jun. 2003.

[3] A. Bykhovskaya and V. Gorin, "Canonical Correlation Analysis: review," arXiv preprint arXiv:2411.15625, Nov. 2024.

[4] J. N. Böhm, P. Berens and D. Kobab, “Attraction-Repulsion Spectrum in Neighbor Embeddings”, Journal of Machine Learning Research, vol. 23, no. 95, pp. 1-32, March 2022.

[5] T. T. Cai and R. Ma, “Theoretical Foundations of t-SNE for Visualizing High-Dimensional Clustered Data”, Journal of Machine Learning Research, vol. 23, no. 391, pp. 13581 – 13634, October 2022.

[6] F. Farahnakian and J. Heikkonen, "A deep auto-encoder based approach for intrusion detection system," in Proc. 20th Int. Conf. Adv. Commun. Technol. (ICACT), Chuncheon, South Korea, 2018, pp. 247–252.

[7] C. Gambella, B. Ghaddar, and J. Naoum-Sawaya, "Optimization problems for machine learning: A survey," European Journal of Operational Research, vol. 290, no. 3, pp. 807–828, May 2021.

[8] H. Hotelling, "Relations between two sets of variates," Biometrika, vol. 28, no. 3/4, pp. 321–377, Dec. 1936.

[9] S. Li, X. Ma, and L. Chen, "Feature dimensionality reduction: a review," Complex & Intelligent Systems, vol. 7, no. 3, pp. 1315–1333, Sep. 2021

[10] J. Kim, J. Shin, F. Chazal, A. Rinaldo, and L. Wasserman, "Homotopy reconstruction via the Čech complex and the Vietoris-Rips complex," in Proc. 36th Int. Symp. Comput. Geom. (SoCG 2020), vol. 164, Leibniz Int. Proc. Informatics (LIPIcs), Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020, pp. 54:1–54:19

[11] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," Foundations and Trends in Machine Learning, vol. 12, no. 4, pp. 307–392, 2019.

[12] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in Proc. 2nd Int. Conf. Learn. Represent. (ICLR), Banff, Canada, 2014, pp. 1–14.

[13] L. McInnes, J. Healy, N. Saul, and L. Großberger, "UMAP: Uniform Manifold Approximation and Projection," J. Open Source Software, vol. 3, no. 29, p. 861, Sep. 2018.

[14] M. Mittal, P. G. J, G. P. M S, R. M. Devadas, L. Ambreen and V. Kumar, "Dimensionality Reduction Using UMAP and TSNE Technique," 2024 Second International Conference on Advances in Information Technology (ICAIT), Chikkamagaluru, Karnataka, India, 2024, pp. 1-5

[15] Subhaditya Mukherjee, «Proxy Attention : Approximating Attention in CNNs using Gradient Based Techniques», Masters Thesis, Department of Artificial Intelligence, University of Groningen, 2023.

[16] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding" Science, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000.

[17] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "Global versus local methods in nonlinear dimensionality reduction," in Proc. 15th Adv. Neural Inf. Process. Syst. (NIPS 2002), vol. 15, MIT Press, 2002, pp. 953–960.

[18] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000.

[19] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," J. Mach. Learn. Res., vol. 9, pp. 2579–2605, Nov. 2008.

[20] H.-T. Wang, J. Smallwood, J. Mourao-Miranda, C. H. Xia, T. D. Satterthwaite, D. S. Bassett, and D. Bzdok, "Finding the needle in a high-dimensional haystack: Canonical correlation analysis for neuroscientists," NeuroImage, vol. 216, p. 116745, Aug. 2020.

[21] Y. Wang, H. Huang, C. Rudin, and Y. Shaposhnik, "Understanding how dimension reduction tools work: An empirical approach to deciphering t-SNE, UMAP, TriMap, and PaCMAP for data visualization," J. Mach. Learn. Res, vol. 22, no. 201, pp. 1–73, 2021.

[22] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," Chemometrics and Intelligent Laboratory Systems, vol. 2, no. 1, pp. 37–52, 1987.

Higher order Deep Unfolding Networks for Compressed Sensing

Georgios Nikolaou

ABSTRACT

Deep Neural Networks (DNNs) have achieved impressive performance in various tasks, including image restoration and compressed sensing (CS). Yet, DNN methods are often treated as black boxes, posing challenges for theoretical analysis. To address this, Deep Unfolding Networks (DUNs) have been introduced, achieving state-of-the-art results while maintaining interpretability. However, they typically rely only on first-order operations, overlooking the benefits of higher-order information for image reconstruction. In our work, we propose a Deep Unfolding Network for analysis sparsity compressed sensing, inspired by the Augmented Lagrangian Method (ALM). By unrolling this iterative optimization scheme, we create an end-to-end trainable network with strong theoretical foundations, interpretability, and improved generalizability. We further enhance DUN capabilities by investigating polynomial expansions of features to capture complex signal patterns. Specifically, we introduce a novel plug-and-play (PNP) higher-order module that integrates seamlessly with established network architectures. Our experiments across multiple benchmark datasets demonstrate that second-order polynomial augmentation yields significant performance improvements, particularly at challenging low sampling rates where we achieve over 3dB PSNR gain compared to baseline methods. The proposed approach not only advances state-of-the-art performance in compressed sensing reconstruction but also provides valuable insights into how higher-order interactions can be effectively incorporated into optimization-based deep learning frameworks.

ADVISORS

Yannis Panagakis, Associate Professor, NKUA

1 INTRODUCTION

1.1 Compressed Sensing

Compressed Sensing (CS), also known as Compressive Sensing, is a powerful signal processing technique that enables the reconstruction of signals from a limited number of measurements. Traditional signal acquisition methods follow the Nyquist-Shannon sampling theorem, which dictates that a signal must be sampled at twice its highest frequency to ensure accurate reconstruction [2, 3]. However, CS challenges this principle by leveraging the inherent sparsity of natural signals to achieve accurate recovery from significantly fewer measurements. Specifically, many natural signals are sparse or compressible in some domain. This means that in the right mathematical transform domain, a large portion of the signal's information can be represented with only a few non-zero coefficients [4, 5, 6, 7]. Consequently, we can store only a fraction of the original signal under some transformation and still achieve an *acceptable* degree of reconstruction. The term *acceptable* here may vary based on the specific application and the level of detail required in the reconstructed signal.

Formally, by leveraging certain properties of natural signals, we can reconstruct the original signal $\mathbf{x} \in \mathbb{R}^n$ from a few (noisy) measurements $\mathbf{y} = \mathbf{A}\mathbf{x} + \boldsymbol{\epsilon} \in \mathbb{R}^m$, where $\mathrm{m} \ll \mathrm{n}$ and $\boldsymbol{\epsilon} \in \mathbb{R}^m$ is additive Gaussian noise. Specifically, we leverage the inherent sparsity of natural signals through analysis sparsity [8]. This involves finding a matrix $\mathbf{W} \in \mathbb{R}^{N \times n}$, with $n \ll N$, called the analysis operator, such that the vector $\mathbf{W}\mathbf{x}$ is sparse. The fraction $\frac{\mathrm{m}}{\mathrm{n}}$ is called the CS Ratio. The sampling matrix can be fixed for application such as Magnetic Resonance Imaging (MRI) and Hyperspectral Imaging [9, 10] or learnable when optimizing the sampling process is of essence, for example when we are dealing with compression [11, 12].

1.2 Deep Unfolding Networks

Model-based approaches to compressed sensing (CS) reconstruction tackle the associated inverse problem, often formulated as a convex optimization problem, using iterative optimization algorithms to converge to a solution. These methods

suffer from handcrafted assumptions about the input (such as the sparsity constraint) and slow convergence [13].

Deep Unfolding Networks (DUNs) combine the advantages of optimization-based methods and deep learning by unfolding iterative algorithms into a finite number of layers. Each layer corresponds to an iteration of an optimization algorithm, with parameters learned through data-driven training. This approach allows for faster convergence and improved performance compared to classical iterative methods [10, 12, 13, 14, 15]. A typical DUN architecture follows the formulation:

$$\mathbf{x}^{(k+1)} = \mathrm{f}_{\boldsymbol{\theta}}^{(k)}\left(\mathbf{x}^{(k)}, \mathbf{y}\right)$$

where $\mathrm{f}_{\boldsymbol{\theta}}^{(k)}$ represents the learned update rule at the k-th iteration. By training the entire unfolding framework end-to-end, DUNs can effectively learn optimal step sizes, thresholds, and transformations for specific tasks such as image restoration, denoising, deblurring, and compressed sensing [10, 11, 12].

DUNs have been widely adopted for solving inverse problems due to their ability to retain the theoretical guarantees of optimization-based methods while leveraging the expressive power of deep learning. Despite their success, existing DUN architectures primarily rely on linear transformations with non-linear activation functions and handcrafted priors, limiting their potential to capture more complex feature interactions. This limitation motivates the exploration of higher-order information in DUNs, which can be achieved using ideas from Polynomial Networks (PNs).

1.3 Polynomial Networks

Polynomial Networks (PNs) extend traditional deep learning architectures by incorporating higher-order feature interactions through polynomial expansions. Unlike standard neural networks that primarily rely on linear transformations followed by activation functions, PNs explicitly model complex dependencies by introducing element-wise multiplicative interactions and benefit from both enhanced expressivity and by interpretable representations [16, 19].

The core idea behind PNs is to augment the feature space by applying polynomial mappings. As described in [18], an N-th order polynomial expansion of input $\mathbf{z} \in \mathbb{R}^{\mathrm{d}}$ is a function $\mathrm{G}: \mathbb{R}^{\mathrm{d}} \to \mathbb{R}^{\mathrm{o}}$ such that:

$$\mathrm{G}(\mathbf{z}) = \boldsymbol{\beta} + \sum_{\boldsymbol{n=1}}^{\boldsymbol{N}} \boldsymbol{\mathcal{W}}_{(1)}^{[n]} \mathbf{z}_{\mathrm{n}}$$

where $\boldsymbol{\beta} \in \mathbb{R}^{\mathrm{o}}$ and $\left\{\boldsymbol{\mathcal{W}}_{(1)}^{[n]} \in \mathbb{R}^{\mathrm{o} \times \mathrm{d}^{\mathrm{n}}}\right\}$ are learnable parameters, and $\mathbf{z}_{\mathrm{n}} = \underbrace{\mathbf{z} \odot \cdots \odot \mathbf{z}}_{n \text{ times}}$.

This formulation enables PNs to capture intricate relationships between input features, improving their ability to generalize and extrapolate beyond training data. Recent studies have demonstrated that PNs achieve state-of-the-art performance in various applications, such as image classification and generation [19, 20].

1.4 Notation

Tensors are denoted by uppercase calligraphic bold-face letters: $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. Matrices (vectors) are denoted by uppercase (lowercase) bold-face letters: $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2}$ ($\mathbf{x} \in \mathbb{R}^{I_1}$). Furthermore the operations of *Hadamard*, *Khatri-Rao* products, as well as the *mode-m unfolding* of a tensor, are defined in [1] as:

- *Hadamard product:* It is the element-wise matrix product. Given matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{\mathrm{I} \times \mathrm{J}}$, their Hadamard product is denoted by $\mathbf{A} * \mathbf{B} \in \mathbb{R}^{\mathrm{I} \times \mathrm{J}}$.
- *Khatri-Rao product:* Given matrices $\mathbf{A} \in \mathbb{R}^{\mathrm{I} \times \mathrm{N}}$ and $\mathbf{B} \in \mathbb{R}^{\mathrm{J} \times \mathrm{N}}$, their Khatri-Rao product is denoted by $\mathbf{A} \odot \mathbf{B} \in \mathbb{R}^{(\mathrm{IJ}) \times \mathrm{N}}$. Let $\mathrm{I} = \mathrm{J} = 2, \mathrm{N} = 3$, then:

$$\mathbf{A} = \begin{pmatrix} \mathrm{a} & \mathrm{b} & \mathrm{c} \\ \mathrm{d} & \mathrm{e} & \mathrm{f} \end{pmatrix} \mathbf{B} = \begin{pmatrix} \mathrm{g} & \mathrm{h} & \mathrm{i} \\ \mathrm{j} & \mathrm{k} & \mathrm{l} \end{pmatrix} \mathbf{A} \odot \mathbf{B} = \begin{pmatrix} \mathrm{ag} & \mathrm{bh} & \mathrm{ci} \\ \mathrm{aj} & \mathrm{bk} & \mathrm{cl} \\ \mathrm{dg} & \mathrm{eh} & \mathrm{fi} \\ \mathrm{dj} & \mathrm{ek} & \mathrm{fl} \end{pmatrix}$$

- *Mode-m unfolding:* Given a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{\mathrm{I}_1 \times \cdots \times \mathrm{I}_{\mathrm{m}} \times \cdots \times \mathrm{I}_{\mathrm{N}}}$, its mode-m unfloding is denoted by $\boldsymbol{\mathcal{X}}_{(\mathrm{m})} \in \mathbb{R}^{\mathrm{I}_{\mathrm{m}} \times (\prod_{\mathrm{n}=1, \mathrm{n} \neq \mathrm{m}}^{\mathrm{N}} \mathrm{I}_{\mathrm{n}})}$ and arranges the mode-m fibers of $\boldsymbol{\mathcal{X}}$ to be the columns of the resulting matrix.

The proximal operator of $\mathbf{v} \in \mathbb{R}^n$ with respect to function $f: \mathbb{R}^n \to \mathbb{R}$ is defined as:

$$\text{prox}_{\lambda f}(\mathbf{v}) = \underset{\mathbf{x} \in \mathbb{R}^n}{\arg\min}\, f(\mathbf{x}) + \frac{1}{2\lambda}\|\mathbf{x} - \mathbf{v}\|_2^2$$

2 BACKGROUND AND RELATED WORK

2.1 Polynomial Networks

Polynomial Networks (PNs) introduce higher-degree polynomial expansions by incorporating element-wise products of features, thereby enabling the capture of complex interactions. By explicitly modeling higher-order dependencies, these networks extend the expressive power of deep learning models beyond traditional architectures.

Early work on polynomial networks demonstrated their potential for function approximation, but their application in modern deep learning has been revitalized through improved training techniques and architectures [16, 17, 18]. Recent studies have demonstrated that PNs are particularly effective for tasks such as image generation and recognition, where feature interactions play a crucial role in representation learning [18, 20].

One key advantage of PNs is their ability to enhance extrapolation capabilities, as polynomial expansions can inherently generalize beyond the training distribution. Additionally, researchers have explored methods to regularize polynomial expansions to prevent overfitting, making them a promising avenue for robust and interpretable deep learning models [19].

2.2 Attention Mechanisms and Higher-Order Information

Attention mechanisms have revolutionized deep learning by dynamically assigning weights to input features, allowing models to focus on the most relevant information. While attention mechanisms were initially developed to capture long-range dependencies, many modern attention modules inherently

leverage higher-order interactions to refine feature representations. Higher-order interactions emerge in attention mechanisms through the computation of pairwise and multi-wise relationships across spatial and channel dimensions. Several variants of attention modules, such as Channel Attention Blocks (CABs) and Self-Attention Mechanisms (SAMs), exploit this property to improve model performance [11, 21, 22]. These modules effectively enhance feature selectivity and enable networks to capture richer contextual information. In image restoration and compressed sensing, attention-based models have demonstrated superior performance by refining spatially and contextually significant details. However, explicit higher-order augmentation strategies beyond traditional attention mechanisms remain largely unexplored.

2.3 Deep Unfolding Networks (DUNs)

Deep neural networks (DNNs) have been at the forefront of computer vision research for the past decade due to their exceptional capabilities in image recognition and various image restoration tasks [23, 24, 25, 26, 27]. The introduction of ResNet [28], which enabled the training of much deeper networks, further complicated the theoretical analysis of DNNs, adding to the already challenging task.

To combine interpretability and adaptivity, some hybrid methods integrate deep neural networks into classic optimization algorithms. Deep unfolding networks (DUNs) [10, 11, 12, 29, 30, 31] have emerged. By unfolding iterative algorithms and interpreting each iteration as the layer of a DNN, they optimize all parameters end-to-end for improved performance and faster convergence. DUNs have excelled in solving ill-posed inverse problems like image deblurring, denoising [11, 12, 14], and Compressed Sensing [10, 12, 15], achieving state-of-the-art performance. However, their growing complexity has made theoretical analysis more challenging.

To our knowledge, no other higher-order augmentation module has been proposed and in the context of image restoration, higher-order information is mainly utilized in attention modules [21, 23].

3 METHODOLOGY

3.1 Setup

We approach Compressed Sensing through *analysis sparsity*, solving the following optimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{W}\mathbf{x}\|_1 \text{ s.t. } \mathbf{A}\mathbf{x} = \mathbf{y}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the original image, $\mathbf{y} \in \mathbb{R}^m$ is the measurement, $\mathbf{A} \in \mathbb{R}^{m\times n}$ is the sampling matrix and $\mathbf{W} \in \mathbb{R}^{N\times n}$ is the analysis operator, with $m \ll n \ll N$.

Due to the non-smoothness of the ℓ_1 norm, we are essentially forced to rely on subgradient methods or similar approaches, which are known to be slow [32]. To address this, we introduce a quadratic penalty for the constraint $\frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2$ and write the problem equivalently as:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \mathrm{f}_\gamma(\mathbf{x}) + \mathrm{g}(\mathbf{W}\mathbf{x})$$

where $\mathrm{f}_\gamma(\mathbf{x}) \coloneqq \frac{\gamma}{2}\|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2$ (γ is a hyper-parameter) and $\mathrm{g}(\mathbf{W}\mathbf{x}) \coloneqq \|\mathbf{W}\mathbf{x}\|_1$.

However, to isolate the non-smooth term, we define an auxiliary variable $\mathbf{z} \in \mathbb{R}^N$ so that $\mathbf{z} = \mathbf{W}\mathbf{x}$. The resulting problem is:

$$\min_{\mathbf{x}\in\mathbb{R}^n, \mathbf{z}\in\mathbb{R}^N} \mathrm{f}_\gamma(\mathbf{x}) + \mathrm{g}(\mathbf{z}) \text{ s.t. } \mathbf{W}\mathbf{x} - \mathbf{z} = \mathbf{0}$$

This formulation allows us to handle each part more efficiently.

3.2 Augmented Lagrangian method (ALM)

A classical approach to solving optimization problems with linear constraints is to convert the *primal* problem into the *dual* problem and solve the dual. Under suitable conditions (e.g., Slater's condition [37]), the primal and dual solutions coincide, and often the dual can be easier to solve or analyze. This is the motivation behind introducing the Lagrangian and discussing primal–dual relationships. We can reformulate the last minimization problem as a min-max problem:

$$\min_{\mathbf{x}\in\mathbb{R}^n,\mathbf{z}\in\mathbb{R}^N} \max_{\boldsymbol{\lambda}\in\mathbb{R}^N} \mathcal{L}(\mathbf{x},\mathbf{z},\boldsymbol{\lambda};\gamma)$$

where $\mathcal{L}(\mathbf{x},\mathbf{z},\boldsymbol{\lambda};\gamma) \coloneqq \mathrm{f}_\gamma(\mathbf{x}) + \mathrm{g}(\mathbf{z}) + \langle\boldsymbol{\lambda}, \mathbf{W}\mathbf{x} - \mathbf{z}\rangle$ is known as the Lagrangian of the problem. Under suitable assumptions, we can interchange the maximization and minimization, thus obtaining the dual solution:

$$\mathbf{d}^* = \max_{\boldsymbol{\lambda}\in\mathbb{R}^N} \left(d(\boldsymbol{\lambda}) \coloneqq \min_{\mathbf{x}\in\mathbb{R}^n,\mathbf{z}\in\mathbb{R}^N} \mathcal{L}(\mathbf{x},\mathbf{z},\boldsymbol{\lambda};\gamma) \right)$$

Nonetheless, the straightforward dual approach still has drawbacks. First $d(\boldsymbol{\lambda})$ is not necessarily smooth, leading to slower convergence rates as before. Second $d(\boldsymbol{\lambda})$ is not necessarily well-defined for all $\boldsymbol{\lambda}$.

To address these issues, one can augment the Lagrangian with a quadratic penalty term, leading to the augmented Lagrangian method. The augmented Lagrangian for this problem is:

$$\mathbf{d}_{\boldsymbol{\beta}}^* = \max_{\boldsymbol{\lambda}\in\mathbb{R}^N} \left(d_\beta(\boldsymbol{\lambda}) \coloneqq \min_{\mathbf{x}\in\mathbb{R}^n,\mathbf{z}\in\mathbb{R}^N} \mathcal{L}_\beta(\mathbf{x},\mathbf{z},\boldsymbol{\lambda};\gamma) \coloneqq \mathcal{L}(\mathbf{x},\mathbf{z},\boldsymbol{\lambda};\gamma) + \frac{\beta}{2}\|\mathbf{W}\mathbf{x} - \mathbf{z}\|_2^2 \right)$$

where $\beta > 0$ is the regularization hyper-parameter.

The augmented dual problem $d_\beta(\boldsymbol{\lambda})$ is convex, smooth and well-defined for all values of $\boldsymbol{\lambda}$, allowing us to apply accelerated gradient methods [33, 34] in the dual!

Algorithm 1: Augmented Lagrangian method (ALM) [33]

1. Choose $\boldsymbol{\lambda} \in \mathbb{R}^N$ and $\beta > 0$
2. **For** $k = 0,1,\ldots$:
   - 2.a Solve $d_\beta(\boldsymbol{\lambda}) \coloneqq \min_{\mathbf{x}\in\mathbb{R}^n,\mathbf{z}\in\mathbb{R}^N} \mathcal{L}_\beta(\mathbf{x},\mathbf{z},\boldsymbol{\lambda})$
   - 2.b Compute $\nabla_{\boldsymbol{\lambda}} d_\beta(\boldsymbol{\lambda})$
   - 2.c Update $\boldsymbol{\lambda}^{\mathrm{k+1}} \coloneqq \boldsymbol{\lambda}^{\mathrm{k}} + \beta\nabla_{\boldsymbol{\lambda}} d_\beta(\boldsymbol{\lambda})$

3.2.1 Solving $d_{\boldsymbol{\beta}}(\boldsymbol{\lambda})$

In practice, this step is carried out by sequential gradient steps—one for $\mathbf{x}$ and one for $\mathbf{z}$—or by applying proximal operators if $\mathrm{g}(\cdot)$ is non-smooth but has an easy proximal form.

To derive the update rule for $\mathbf{x}$, we employ the majorization-minimization (MM) framework [35]. Rather than minimizing the original objective directly, we minimize a surrogate that upper bounds the function and is easier to optimize. In this case, the surrogate takes the form of a quadratic majorant of the augmented Lagrangian, leading to a standard gradient descent step as derived in [35]. The first step in this process is computing the partial derivative of $\mathcal{L}_{\beta}(\mathbf{x}, \mathbf{z}, \boldsymbol{\lambda}; \gamma)$ with respect to $\mathbf{x}$:

$$\frac{\partial \mathcal{L}_{\beta}(\mathbf{x}, \mathbf{z}, \boldsymbol{\lambda}; \gamma)}{\partial \mathbf{x}} = \gamma \mathbf{A}^{\top}(\mathbf{A}\mathbf{x} - \mathbf{y}) + \beta \mathbf{W}^{\top}\left(\mathbf{W}\mathbf{x} - \mathbf{z} + \frac{1}{\beta}\boldsymbol{\lambda}\right)$$

Then, we just update $\mathbf{x}$ by doing a simple gradient descent step, namely:

$$\begin{aligned}\mathbf{x}^{\mathrm{k}+1} &= \mathbf{x}^{\mathrm{k}} - \rho \frac{\partial \mathcal{L}_{\beta}\left(\mathbf{x}, \mathbf{z}^{\mathrm{k}}, \boldsymbol{\lambda}^{\mathrm{k}}; \gamma\right)}{\partial \mathbf{x}} \\ &= \mathbf{x}^{\mathrm{k}} - \rho\left[\gamma \mathbf{A}^{\top}(\mathbf{A}\mathbf{x} - \mathbf{y}) + \beta \mathbf{W}^{\top}\left(\mathbf{W}\mathbf{x} - \mathbf{z}^{\mathrm{k}} + \frac{1}{\beta}\boldsymbol{\lambda}^{\mathrm{k}}\right)\right]\end{aligned}$$

where $\rho = \frac{1}{L}$ is the learning rate and L is the Lipschitz constant of $\mathcal{L}_{\beta}(\cdot, \mathbf{z}, \boldsymbol{\lambda}; \gamma)$.

To derive the update rule for $\mathbf{z}$, we observe that, unfortunately, the augmented Lagrangian is not differentiable in $\mathbf{z}$ due to the non-smooth ℓ_1-norm term. However, the ℓ_1-norm admits a tractable proximal operator [35], which we can exploit to perform the update. Specifically, we aim to solve:

$$\begin{aligned}\mathbf{z}^{\mathrm{k}+1} &= \min_{\mathbf{z} \in \mathbb{R}^N} \mathcal{L}_{\beta}\left(\mathbf{x}^{\mathrm{k}+1}, \mathbf{z}, \boldsymbol{\lambda}^{\mathrm{k}}\right) \\ &= \min_{\mathbf{z} \in \mathbb{R}^N} \mathrm{g}(\mathbf{z}) + \left\langle \boldsymbol{\lambda}^{\mathrm{k}}, \mathbf{W}\mathbf{x}^{\mathrm{k}+1} - \mathbf{z} \right\rangle + \frac{\beta}{2}\left\|\mathbf{W}\mathbf{x}^{\mathrm{k}+1} - \mathbf{z}\right\|_2^2 \\ &= \min_{\mathbf{z} \in \mathbb{R}^N} \mathrm{g}(\mathbf{z}) + \frac{\beta}{2}\left\|\mathbf{W}\mathbf{x}^{\mathrm{k}+1} - \mathbf{z} + \frac{1}{\beta}\boldsymbol{\lambda}^{\mathrm{k}}\right\|_2^2\end{aligned}$$

$$= \mathrm{prox}_{ag}\left(\mathbf{W}\mathbf{x}^{\mathrm{k}+1} + \frac{1}{\beta}\boldsymbol{\lambda}^{\mathrm{k}}\right)$$

$$= \mathrm{S}_{a}\left(\mathbf{W}\mathbf{x}^{\mathrm{k}+1} + \frac{1}{\beta}\boldsymbol{\lambda}^{\mathrm{k}}\right)$$

where $a = \frac{1}{\beta}$ and $\mathrm{S}_a(\mathbf{y}) = \mathrm{sign}(\mathbf{y})\max(0, |\mathbf{y}| - a)$ is the Soft-Thresholding operator, applied element-wise to $\mathbf{y}$.

3.2.2 Updating the dual parameter λ

Since $d_\beta(\boldsymbol{\lambda})$ is an implicitly defined function of $\boldsymbol{\lambda}$, we cannot directly compute the partial derivative of it with respect to $\boldsymbol{\lambda}$. But under some mild uniqueness conditions, an immediate consequence of Danskin's Theorem [36] is the following:

$$\nabla_{\boldsymbol{\lambda}} d_\beta(\boldsymbol{\lambda}) = \frac{\partial \mathcal{L}_\beta\left(\mathbf{x}^{\mathrm{k}+1}, \mathbf{z}^{\mathrm{k}+1}, \boldsymbol{\lambda}; \gamma\right)}{\partial \boldsymbol{\lambda}} = \mathbf{W}\mathbf{x}^{\mathrm{k}+1} - \mathbf{z}^{\mathrm{k}+1}$$

Therefore, the update rule for $\boldsymbol{\lambda}$ is a simple gradient ascent step:

$$\boldsymbol{\lambda}^{\mathrm{k}+1} = \boldsymbol{\lambda}^{\mathrm{k}} + \beta \nabla_{\boldsymbol{\lambda}} d_\beta(\boldsymbol{\lambda})$$

$$= \boldsymbol{\lambda}^{\mathrm{k}} + \beta\left[\mathbf{W}\mathbf{x}^{\mathrm{k}+1} - \mathbf{z}^{\mathrm{k}+1}\right]$$

3.3 Proposed Deep Unfolding Network

Given the updates rule discussed in Section 3.2, we can directly construct the neural network, by allowing $\mathbf{A}$ and $\mathbf{W}$ to be a learnable parameters, and unfolding the iterative scheme for a fixed number of iterations.

Specifically, the network, dubbed *DUNet* (Deep Unfolding Network), takes as input a measurement $\mathbf{y} \in \mathbb{R}^m$ and initializes the iterates as:

$$\mathbf{x}^0 = \mathbf{A}^\top \mathbf{y}, \qquad \mathbf{z}^0 = \boldsymbol{\lambda}^0 = \mathbf{0}_N$$

The network has L layers, where each layer performs one step of Algorithm 1. The sampling matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ is naturally shared across all layers, while layer d has its own analysis operator $\mathbf{W}^{(d)} \in \mathbb{R}^{N \times n}$, making the network more expressive. Finally, the network returns a truncated $\mathbf{x}^L$ using a clipping function that is

defined for pairs of $(\mathbf{x}, \mathbf{y})$ as $\psi(\mathbf{x}, \mathbf{y}) = \max(\boldsymbol{y}_{\mathbf{min}}, \min(\mathbf{y}_{\mathbf{max}}, \mathbf{x}))$, applied element-wise. The truncation is performed to ensure that the output is on the same range as the input.

3.4 Higher-order Block

We introduce a novel module, called Polynomial Block, designed to leverage higher-order information and enhance fine-detail capture in input signal reconstruction. A polynomial block of order n can be expressed mathematically:

$$\mathrm{G}^{\mathrm{n}}(\mathbf{x}) = \begin{cases} \psi\left(\left(\mathbf{A}^{(n)}\mathbf{x}\right) * \left(\mathbf{B}^{(n)}\mathrm{G}^{\mathrm{n}-1}(\mathbf{x})\right), \mathbf{y}\right), & \boldsymbol{n} \geq \mathbf{2} \\ \mathbf{x} + \mathbf{A}^{(\mathbf{1})}\mathbf{x} & , \quad \boldsymbol{n} = \mathbf{1} \end{cases}$$

where matrices $\mathbf{A}^{(\mathrm{n})}, \mathbf{B}^{(\mathrm{n})}$ are learnable parameters.

3.4.1 Augmenting DUNet

After carefully examining the placement of augmentation, we introduced it immediately following the $\mathbf{x}$-update for two reasons. First, it allows us to better leverage the nature of the optimization algorithm. While the $\mathbf{x}$-update rule is derived strictly from an algorithmic perspective and is essential for reconstruction, the $\mathbf{z}$ and $\mathbf{u}$ updates are auxiliary, providing greater flexibility in their inputs. Second, higher-order information can be utilized when updating $\mathbf{z}$ and $\mathbf{u}$ without increasing the number of parameters, thereby enhancing the algorithm's expressive power without the extra cost.

3.4.2 Augmenting ISTA-Net+

ISTA-Net [10], short for Iterative Shrinkage-Thresholding Algorithm, has made significant contributions to compressed sensing by incorporating the Proximal Gradient Descent algorithm [35]. Unlike our approach, ISTA-Net employs a data-driven method to learn the transformation to the sparse domain. After the gradient descent step, it applies convolutions and activation functions both before and after the soft-thresholding operator. Following a rationale similar to that of the DUNet augmentation, we introduce a second-order polynomial block just before the soft-thresholding operator.

4 EVALUATION

To distinguish between models, when an augmentation of degree N is added, we append to the name of the model the suffix "$/\mathrm{A}N$".

4.1 Datasets

- **BSD400 [39]:** The dataset consists of 400 natural images, covering a diverse range of scenes and objects. The exact dimensions of each image can differ, but they are often in the range of a few hundred pixels in both width and height.
- **BSD68 [39]:** A smaller version of BSD400, also consisting of similar-sized natural images. Often used for quicker evaluation and testing for both inverse and segmentation problems.
- **Set11 [38]:** A small dataset, containing just 11 images of scenes and objects, often employed as a quick test for various image processing task due to its simplicity and small size.

4.2 Training Configuration

4.2.1 DUNet

The network is trained end-to-end using the widely adopted BSD400 dataset. Similar to [12], we extract 32×32 patches from this dataset, applying random rotations and flips for data augmentation. Model evaluation is conducted on Set11 and BSD68 datasets, and performance is assessed using the PSNR and SSIM metrics [40].

We employ the Adam optimizer with a learning rate of $5\text{e-}4$, with no weight decay, and set the batch size to 128, the redundancy ratio to 4 and we do not utilize a learning rate scheduler. We unfold the optimization algorithm for 10 iterations and initialize the (learnable) parameters γ and β to $1\text{e-}3$ and $1\text{e-}2$ respectively, training the network for 100 epochs.

4.2.2 ISTA-Net+

Our baseline model is ISTA-Net+. We use the parameters and implementation from the original paper for its training. Additionally, we augment the network with our proposed module. Both models are evaluated on the Set11 and BSD68 datasets.

Specifically, the network has 9 layers and is trained using the Adam optimizer with a learning rate of $1e\text{-}4$, no weight decay and scheduler. Vanilla ISTA-Net+ is trained for 200 epochs (as it does in the original paper), while the augmented version is trained for 50 epochs.

4.3 Compressed Sensing Results

Dataset	DUNet	DUNet/A1	DUNet/A2
Set11	31.20 / 0.9163	33.36 / 0.9408	33.80 / 0.9421
BSD68	29.69 / 0.8858	30.89 / 0.9033	30.89 / 0.8951

Table 1: Ablation study of polynomial degree. Fixed 25% CS Ratio. The reported metrics are PSNR/SSIM

We begin by validating the utility of the augmentation. Table 1 presents the performance of DUNet and its augmentations. It is evident that introducing a first-order augmentation significantly improves the performance of the original model by enhancing its expressiveness.

Furthermore, adding a second-order augmentation provides an additional performance boost on the Set11 dataset, but not on the BSD68, suggesting that increasing the polynomial degree further will not yield a performance boost. Based on these results, we adopt a second-order augmentation for further experiments.

Dataset	CS Ratio	ISTA-Net+	ISTA-Net+/A2	DUNet/A2
	1%	17.71 / 0.4268	17.78 / 0.4303	**20.94 / 0.5411**
	4%	21.83 / 0.6261	22.23 / 0.6492	**25.39 / 0.7779**

Set11	10%	26.97 / 0.8181	<u>27.33 / 0.8289</u>	**29.14 / 0.8787**
	25%	32.89 / 0.9285	<u>33.24 / 0.9330</u>	**33.80 / 0.9421**
	50%	38.51 / 0.9720	<u>38.77 / 0.9731</u>	**38.89 / 0.9764**
BSD68	1%	19.43 / 0.4279	<u>19.46 / 0.4310</u>	**22.00 / 0.5141**
	4%	22.61 / 0.5689	<u>22.78 / 0.5766</u>	**24.97 / 0.6801**
	10%	25.62 / 0.7153	<u>25.77 / 0.7216</u>	**27.24 / 0.7941**
	25%	29.63 / 0.8596	<u>29.82 / 0.8649</u>	**30.89 / 0.8951**
	50%	34.54 / 0.9479	<u>34.56 / 0.9492</u>	**35.82 / 0.9666**

Table 2: Results of image compressed sensing. The best and second-best scores are highlighted and underlined respectively. Metrics: PSNR/SSIM

Table 2 presents a quantitative comparison of the models. The augmented ISTA-Net+ consistently outperforms its vanilla counterpart across both datasets and metrics. Additionally, despite an increase in per-epoch training time, the overall training time decreases by 25%. This indicates that the augmentation is beneficial for both performance and time complexity.

Figure 1: Visual comparison of ablation study models with the CS ratio being 10%

Our augmented DUNet surpasses both ISTA-based models. For lower CS ratios, we observe significant improvements in the PSNR metric compared to ISTA-Net+ and ISTA-Net+/A2, exceeding 3 dB on the Set11 dataset and 2 dB on the BSD48 dataset. The superiority of our model at lower CS ratios is further demonstrated in the visual comparison in Figure 1, supporting our earlier claim that the network effectively captures intricate image features and leverages them to enhance performance.

For larger CS ratios, the performance gap between ISTA-based models and DUNet/A2 narrows. However, our proposed model still surpasses the baseline models both quantitatively and qualitatively, as shown in Figure 2.

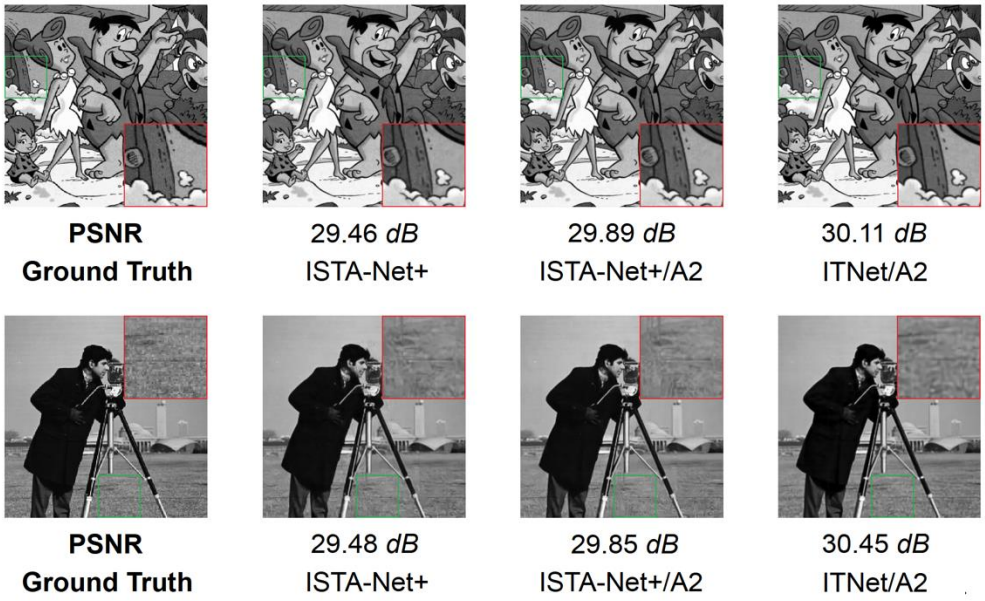


Figure 2: Visual comparison of ablation study models with the CS ratio being 25%

5 CONCLUSION

In this work, we focus on integrating higher-order information into Deep Unfolding Networks. The proposed higher-order augmentation offers several advantages: it enhances the model's expressivity and, in some cases, reduces training time without compromising performance. However, for polynomial

degrees of three or higher, the added computational cost outweighs the performance gains, making such configurations impractical.

The core novelty of the augmentation lies in its design as a self-contained, architecture-agnostic module that can be treated as a black box and seamlessly integrated into existing models in a plug-and-play (PnP) fashion. Although its placement within the model architecture requires some consideration, it imposes no constraints on the surrounding design and introduces minimal additional complexity. Beyond its flexibility, it provides a lightweight yet effective means of increasing the model's representational capacity, enabling improved performance without significantly altering the original architecture. We validate its effectiveness through comprehensive experiments on widely used datasets across a range of CS ratios, consistently demonstrating significant improvements over baseline models.

While the current augmentation retains some degree of interpretability, further improvements may be achieved by removing activation functions—a direction that warrants exploration in future work. Drawing deeper inspiration from polynomial networks may also yield insights into model behavior and contribute to the development of more interpretable, state-of-the-art architectures. Additionally, future research should investigate generalization error and other theoretical properties to address the "black box" nature often associated with deep neural networks.

REFERENCES

[1] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. SIAM Review, 51(3):455–500, 2009.

[2] H. Nyquist. Certain topics in telegraph transmission theory. Transactions of the American Institute of Electrical Engineers, 47(2):617–644, 1928.

[3] C.E. Shannon. Communication in the presence of noise. Proceedings of the IRE, 37(1):10–21, 1949.

[4] E.J. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Transactions on Information Theory, 52(2):489–509, 2006.

[5] Anna C. Gilbert, Piotr Indyk, Mark Iwen, and Ludwig Schmidt. Recent developments in the sparse fourier transform: A compressed fourier transform for big data. IEEE Signal Processing Magazine, 31(5):91–100, 2014.

[6] S.G. Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. IEEE Transactions on Signal Processing, 41(12):3397–3415, 1993.

[7] Ingrid Daubechies. Ten Lectures on Wavelets. Society for Industrial and Applied Mathematics, 1992.

[8] David L. Donoho and Michael Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via minimization. Proceedings of the National Academy of Sciences, 100(5):2197–2202, 2003

[9] Jong Chul Ye. Compressed sensing MRI: a review from signal processing perspective. BMC Biomed. Eng., 1(1), December 2019.

[10] Jian Zhang and Bernard Ghanem. ISTA-net: Interpretable optimization-inspired deep network for image compressive sensing. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. IEEE, June 2018.

[11] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-Stage Progressive Image Restoration. In 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, June 2021.

[12] Chong Mou, Qian Wang, and Jian Zhang. Deep generalized unfolding networks for image restoration. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, June 2022.

[13] Bin Chen, Jiechong Song, Jingfen Xie, and Jian Zhang. Deep physics-guided unrolling generalization for compressed sensing. Int. J. Comput. Vis., 131(11):2864-2887, November 2023

[14] Jian Zhang, Chen Zhao, and Wen Gao. Optimization-inspired compact deep compressive sensing. IEEE J. Sel. Top. Signal Process., 14(4):765–774, May 2020.

[15] Zhonghao Zhang, Yipeng Liu, Jiani Liu, Fei Wen, and Ce Zhu. AMP-Net: Denoising-based deep unfolding for compressive image sensing. IEEE Trans. Image Process., 30:1487–1500, 2021.

[16] Moulik Choraria, Leello Tadesse Dadi, Grigorios Chrysos, Julien Mairal, and Volkan Cevher. The spectral bias of polynomial neural networks. In International Conference on Learning Representations, 2022.

[17] Grigorios G. Chrysos, Markos Georgopoulos, Jiankang Deng, Jean Kossaifi, Yannis Panagakis, and Anima Anandkumar. Augmenting deep classifiers with polynomial neural networks. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, Computer Vision – ECCV 2022, pages 692–716, Cham, 2022. Springer Nature Switzerland.

[18] Grigorios G. Chrysos, Stylianos Moschoglou, Giorgos Bouritsas, Jiankang Deng, Yannis Panagakis, and Stefanos Zafeiriou. Deep polynomial neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(8):4021–4034, 2022.

[19] Grigorios G. Chrysos, Bohan Wang, Jiankang Deng, and Volkan Cevher. Regularization of polynomial networks for image recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 16123–16132, June 2023.

[20] Grigorios Chrysos, Markos Georgopoulos, and Yannis Panagakis. Conditional generation using polynomial expansions. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, Advances in Neural Information Processing Systems, volume 34, pages 28390–28404. Curran Associates, Inc., 2021.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S.

Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017.

[22] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. CBAM: Convolutional Block Attention Module. In Computer Vision – ECCV 2018, Lecture notes in computer science, pages 3–19. Springer International Publishing, Cham, 2018.

[23] Weisheng Dong, Peiyao Wang, Wotao Yin, Guangming Shi, Fangfang Wu, and Xiaotong Lu. Denoising prior driven deep neural network for image restoration. IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(10):2305–2318, 2019.

[24] Victor Lempitsky, Andrea Vedaldi, and Dmitry Ulyanov. Deep image prior. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 9446–9454, 2018.

[25] Zhendong Wang, Xiaodong Cun, Jianmin Bao, Wengang Zhou, Jianzhuang Liu, and Houqiang Li. Uformer: A general u-shaped transformer for image restoration. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 17662–17672, 2022.

[26] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.

[27] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image restoration. TPAMI, 2020.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778, 2016

[29] Kai Zhang, Luc Van Gool, and Radu Timofte. Deep unfolding network for image super-resolution. In IEEE Conference on Computer Vision and Pattern Recognition, pages 3217–3226, 2020.

[30] C Bertocchi, E Chouzenoux, M-C Corbineau, J-C Pesquet, and M Prato. Deep unfolding of a proximal interior point method for image restoration. Inverse Problems, 36(3):034005, feb 2020.

[31] Vicky Kouni and Yannis Panagakis. DECONET: An unfolding network for analysis-based compressed sensing with generalization error bounds. IEEE Trans. Signal Process., 71:1938–1951, 2023.

[32] Ohad Shamir and Tong Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In Sanjoy Dasgupta and David McAllester, editors, Proceedings of the 30th International Conference on Machine Learning, volume 28 of Proceedings of Machine Learning Research, pages 71–79, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[33] Bingsheng He and Xiaoming Yuan. On the acceleration of augmented Lagrangian method for linearly constrained optimization. 2010.

[34] Yurii Nesterov. Introductory lectures on convex optimization: A basic course, volume 87. Springer, 2004.

[35] Neal Parikh and Stephen P. Boyd. Proximal Algorithms. Found. Trends Optim., 1:127–239, 2013

[36] John M. Danskin. The theory of max-min, with applications. SIAM Journal on Applied Mathematics, 14(4):641–664, 1966

[37] M. Slater. Lagrange Multipliers Revisited. Cowles Commission Discussion Paper No. 403, November, 1950

[38] Kuldeep Kulkarni, Suhas Lohit, Pavan Turaga, Ronan Kerviche, and Amit Ashok. ReconNet: Non-iterative reconstruction of images from compressively sensed measurements. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, June 2016.

[39] D Martin, C Fowlkes, D Tal, and J Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001. IEEE Comput. Soc, 2002.

[40] Zhou Wang, Alan Conrad Bovik, Hamid Rahim Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process., 13(4):600–612, April 2004.

.


Διπλωματικές Εργασίες

Evaluation of Three Text-to-Image Generative AI Models

Evgenios Mazarakis

ABSTRACT

This study investigates the performance of three text-to-image generative models: DALL-E 2, Stable Diffusion, and Pix-Art-α. The evaluation begins with an overview of the selected models, followed by the identification of two key hyperparameters — prompt type, image resolution— which serve as the primary variables for analysis. The prompt parameter is derived from a proposed benchmark, while the image resolution is standardized to (512, 512) and (1024, 1024) pixels, respectively. To facilitate comprehensive evaluation, a multi-task benchmark is designed, encompassing six distinct task categories, each comprising 12 prompts. Furthermore, two complementary evaluation methodologies are employed: human evaluation and the CLIP score metric. This dual evaluation strategy ensures a balanced qualitative and quantitative assessment of the models' performance. The evaluation indicates that the commercial model DALL-E 2 outperforms the free models in both qualitative and quantitative assessments. Notably, the analysis reveals a general deficiency in text-related tasks. Additionally, human features, such as hands, fingers, and eye pupils, are often inadequately represented in the generated images.

Keywords: Generative AI, Text-to-Image Models, Graphics.

ADVISORS

Theoharis Theoharis, Professor NKUA

1 INTRODUCTION

The primary aim of this study is to evaluate the performance of three well-known text-to-image generative models — DALL-E 2, Stable Diffusion, and Pix-Art-α — by examining the influence of two key hyperparameters: prompt type and image resolution. The objective is to determine which model demonstrates superior performance across a standardized benchmark and to assess how image resolution impacts the overall output quality. The central research question assumes that significant performance differences will emerge, driven by the substantial variations in the underlying deep learning architectures and training datasets of the models, thereby necessitating an in-depth investigation.

The evaluation process incorporates min-max normalization [1] to process human evaluation scores and employs the CLIP score metric [2] for quantitative performance assessment.

Results indicate that the commercial model, DALL-E 2, consistently outperforms the open-source models, Stable Diffusion and Pix-Art-α, in both qualitative and quantitative evaluations. However, the analysis highlights a common weakness across all models in handling text generation tasks. Moreover, the generated images frequently exhibit inaccuracies when depicting human features, particularly hands, fingers, and eye pupils.

2 ASSESSING OF TEXT-TO-IMAGE GEN AI MODELS

This study evaluates three well-known text-to-image generative models: the open-source models Stable Diffusion and Pix-Art-α, along with the commercial model DALL-E 2. The assessment focuses on two key hyperparameters: prompt type and image resolution. To support a comprehensive evaluation, a multitask benchmark is introduced, comprising six distinct task categories, each containing twelve predefined prompts. Two complementary evaluation approaches are applied. The first is a qualitative assessment, in which human raters evaluate the generated images, scoring the alignment between each image and its corresponding textual prompt on a scale from 1 (lowest) to 5 (highest). The

second approach is a quantitative evaluation, utilizing the CLIP model to compute similarity scores between each prompt and its corresponding generated image, thereby providing an objective measure of each model's performance.

2.1 Assessing Models

Examining the performance of Generative AI (GAI) models, such as DALL-E 2, Stable Diffusion, and Pix-Art-α, provides valuable insights into their respective capabilities and limitations.

DALL-E 2 [3], developed by OpenAI, is a generative model capable of producing images from textual descriptions. In this study, images are generated using a custom Python script[1] that interacts with the OpenAI API [4].

Stable Diffusion2 [5] is a text-to-image latent diffusion model created through collaborative efforts between CompVis, Stability AI, and LAION. Specifically, this study employs the Stable-Diffusion-v1-5 checkpoint, initialized using weights from the Stable Diffusion-v1-2 checkpoint [5,6].

Pix-Art-α [7] is an open-source text-to-image generation model developed by Huawei Noah's Ark Lab. For this evaluation, images are generated using the PixArt-XL-2-1024-MS checkpoint [8] within the official web application [9], following the predefined benchmark criteria.

2.2 Benchmarking Tasks

This section presents the multi-task benchmark developed to evaluate the performance of text-to-image models. The benchmark is designed to assess each model's ability to interpret and generate various elements described within textual prompts. Its design is inspired by the DrawBench framework [10]. The proposed benchmark encompasses six distinct task categories (coloring,

[1] https://github.com/EMazarakis/Generative-AI-in-Graphics/blob/main/DALL-E%202/PythonApplication5.py

2 https://github.com/EMazarakis/Generative-AI-in-Graphics/blob/main/Stable%20Diffusion/Installation_Instructions.md

counting, conflicting, text, positional, faces) [11], which are detailed in Tables 1, 2, and 3.

Table 3. Coloring and Counting benchmark tasks.

Coloring	Counting
A red colored car.	One tennis ball on the court.
A black colored car.	Three people crossing the street.
A pink colored car.	Two monkeys eating banana.
A navy-blue colored car.	Bench in a park with two backpacks on it.
A red car and a white sheep.	Nine children doing a circle dance around a Christmas tree.
A blue bird and a brown bear.	An office desk with six laptops on it.
A green apple and a black backpack.	Person holding four toy pyramids.
A green cup and a blue cell phone.	Five photograph prints are hanging to dry in a dark room.
A red pencil in a green cup on a blue table.	Person carrying a stack of twelve books.
An office with five desks and seven colorful chairs.	Two people juggling ten balls together.
An orange bird scaring a blue scarecrow with a red pirate hat.	Birthday cake with exactly twenty candles on it.
Two pink football balls and three green basketball balls on a bench.	Office room with fifteen chairs.

Table 2. Conflicting and Text benchmark tasks.

Conflicting	Text
A zebra without strips.	A sign that says 'Hello'.
A penguin in a city.	A sign that says 'World'.
Rainbow colored Ferrari.	A sign that says 'Hello World'.
A giraffe kissing a monkey.	A sign that says 'World Hello'.
A panda inside a cup of tea.	A sign that says 'Speed limit 45'.
A giraffe inside a car.	A sign that says 'Don’t pass !'.

An ant under the sea.	A sign that says 'No pedestrians !'.
A motorcycle inside an oven.	A sign that says 'No overtaking !'.
A polar bear on the desert.	A sign that says 'No Goods materials'.
A man walking on the ceiling.	A sign that says 'No hazardous materials'.
A fish eating a pelican.	A sign that says 'You mustn’t turn Right'.
A horse riding an astronaut in the forest.	A sign that says 'Max speed limit 80 km'.

Table 3. Positional and Faces benchmark tasks.

Positional	Faces
A train on top of a surfboard.	Face of a man with a goatee.
A wine glass on top of a dog.	Face of an angry teacher.
A bicycle on top of a boat.	Face of a kid with a tiger make-up.
An umbrella under a spoon.	Face of an old lady with blue hair and a wide smile.
A car on the left of a bus.	A bald man doing a handstand.
A black apple on the right of a green backpack.	Face of a woman with brown hair looking over her shoulder.
A carrot on the left of a broccoli.	Sad face of a blonde girl holding a popped balloon.
A pizza on the right of a suitcase.	Face of a weightlifting white hair Olympic gold medalist lifting 120kg with a referee next to it encouraging him.
A cat on the right of a tennis racket.	The face of a soldier with camouflaged face painting obscured by leaves.
A stop sign on the right of a refrigerator.	A face of a man with a beard with water splashing onto his face.

A sheep to the right of a wine glass.	A dark messy hair boy rolling his tongue with blue light shinning on his face.
A zebra to the right of a fire hydrant.	A young kid with dark eye bags standing in front of her grandma with wrinkles looking at the sky.

2.3 Assessment Techniques

Two complementary methods are employed to evaluate the performance of the generative AI models. The first is a qualitative evaluation, which relies on human assessments of the generated images. The second is a quantitative evaluation, based on the calculation of the CLIP score.

In the qualitative evaluation, human raters assess image quality according to multiple criteria, including color accuracy, object count, the presence of conflicting scenarios, text presence and readability, facial appearance, and spatial relationships. This evaluation process follows established human rating protocols [11,12]. For each prompt, raters are shown images generated by DALL-E 2, Stable Diffusion, and Pix-Art-α. Each rater answers a single question, rating how well the image aligns with the given textual prompt using a 5-point scale, where 1 indicates the weakest alignment and 5 indicates the strongest alignment [11]. The collected scores were aggregated and subsequently converted into normalized percentage values. Two separate questionnaires were distributed, each containing images grouped by resolution: one for 512x512 and the other for 1024x1024. The question posed was:

- How well does the image represent the text caption: [Text Caption]? [12] — The question subjectively evaluates image-text alignment.

The quantitative evaluation of generated images is conducted using metric-based scores. One widely used metric is the CLIP Score, which measures the semantic similarity between a generated image and its corresponding textual prompt. The primary objective of CLIP [2] is to enhance models' understanding

of the relationship between visual content and textual descriptions. In this study, the CLIP Score is computed using a Python script[3] developed based on the official OpenAI repository [13].

2.4 Results

This section presents the results obtained from the qualitative and quantitative evaluations conducted on the three text-to-image models: DALL-E 2, Stable Diffusion, and Pix-Art-α.

Qualitative assessment results. A survey was conducted with 17 participants to evaluate the performance of the three generative text-to-image models. Each participant reviewed a total of 432 images, assigning a rating on a scale from 1 (lowest) to 5 (highest). To standardize the human evaluation scores, the min-max normalization technique was applied, scaling the ratings to a range of [0, 100]. The formula used for this process is as follows:

$$f = \frac{EvaluationValue \ - \ MinEvaluationValue}{MaxEvaluationValue \ - \ MinEvaluationValue}$$

The resulting values were converted into percentages by multiplying them by 100, yielding normalized ratings (%) for the human evaluation. The data was then organized according to model, image resolution, and prompt type for further analysis[4].

The chart, specifically for 512x512 image resolution, indicates that DALLE-2 has the best image-text alignment compared to the other two models (see Fig. 1).

[3] https://github.com/EMazarakis/Generative-AI-in-Graphics/tree/main/CLIP
[4] https://github.com/EMazarakis/Generative-AI-in-Graphics/tree/main/Human%20Evaluation

Normalized ratings (%) for human evaluation by Task and Model

Task Type	DALL-E 2	Pix-Art-α	Stable Diffusion
Colouring	74,51	67,28	60,17
Conflicting	49,75	61,76	34,44
Counting	74,02	51,47	41,91
Faces	74,02	69,12	52,08
Positional	41,30	48,53	40,44
Text	38,60	13,24	13,73

Fig. 1. Normalized ratings (%) for human evaluation of 512x512 image resolution. Emphasize the highest value in red.

For 1024x1024 image resolution, the chart indicates that DALL-E 2 achieves the highest image-text alignment compared to the other two models (see Fig. 2).

Normalized ratings (%) for human evaluation by Task and Model

Task Type	DALL-E 2	Pix-Art-α	Stable Diffusion
Colouring	72,06	64,22	31,74
Conflicting	50,00	58,46	27,33
Counting	71,08	56,37	31,37
Faces	75,49	72,06	27,08
Positional	46,94	42,28	25,37
Text	49,02	15,44	13,11

Fig. 2. Normalized ratings (%) for human evaluation of 1024x1024 image resolution. Emphasize the highest value in red.

Quantitative assessment results. The CLIP Score evaluates the similarity by computing the cosine similarity between the embeddings generated by the model for both the image and its text description. The ViT-L/14@336px model [2] was selected for its superior performance compared to other models of the CLIP

family. Additionally, the mean and median percentages of the CLIP scores were calculated, with the data organized by model, image resolution, and prompt type.

Reviewing the chart of mean and median values for each task, particularly for 512x512 image resolution, DALL-E 2 shows better image-text alignment than the other models (see Fig. 3).

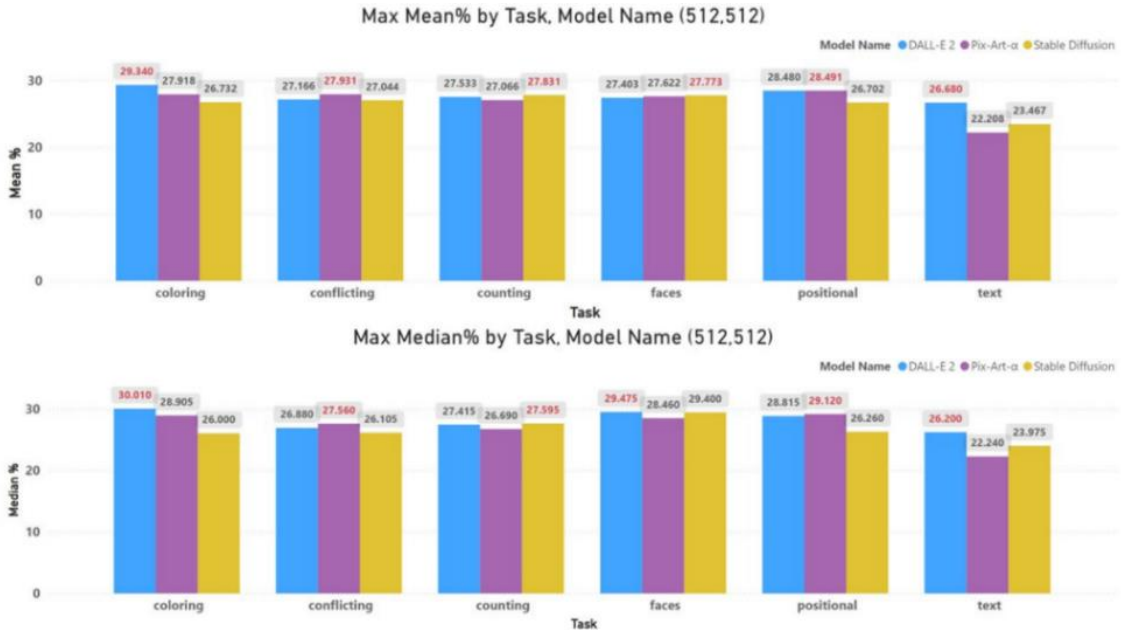


Fig. 3. Mean and Median (%) for 512x512 image resolution by task and model based on ViT-L/14@336px. The highest value in each cluster is highlighted in red.

Reviewing the chart of mean and median values for each task, particularly for 1024x1024 image resolution, DALL-E 2 demonstrates better image-text alignment than the other models (see Fig. 4).

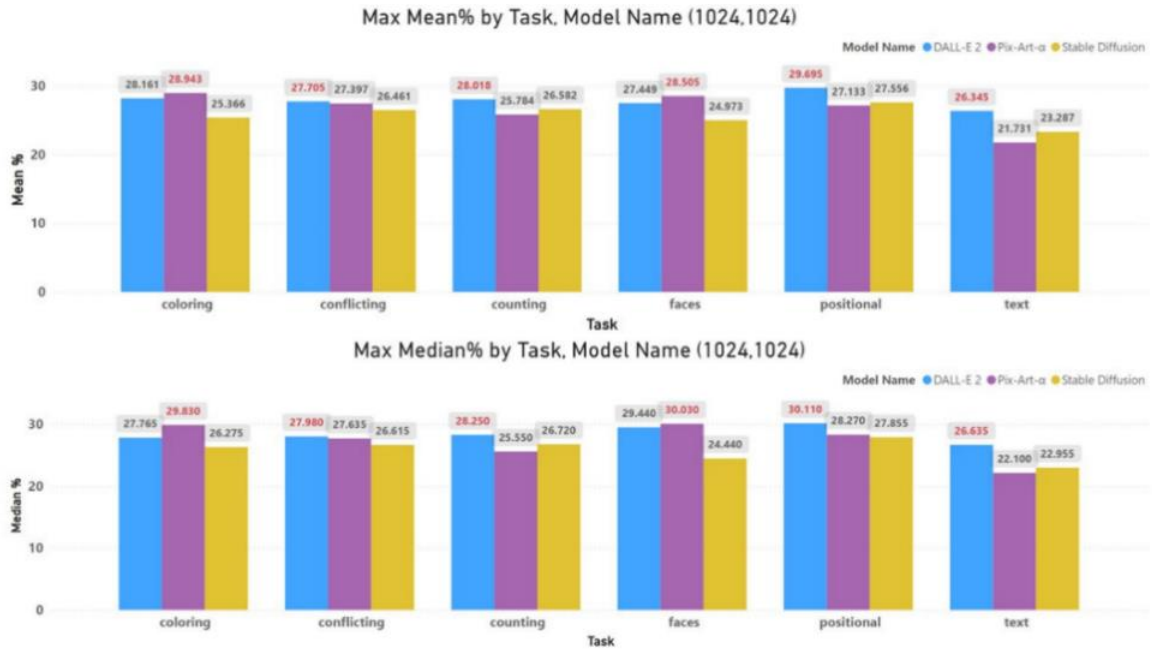


Fig. 4. Mean and Median (%) for 1024x1024 image resolution by task and model based on ViT-L/14@336px. The highest value in each cluster is highlighted in red.

Shortcomings. Some of the models' limitations in various benchmarking tasks are illustrated in the following images.

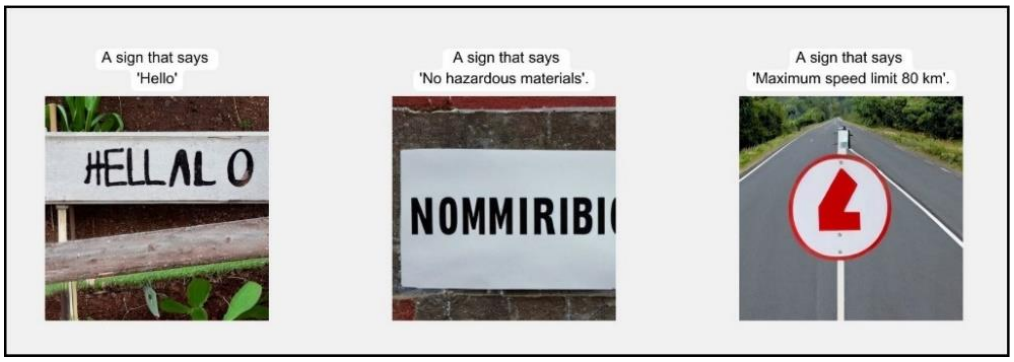


Fig. 5. Text limitations of Stable Diffusion for (512,512).

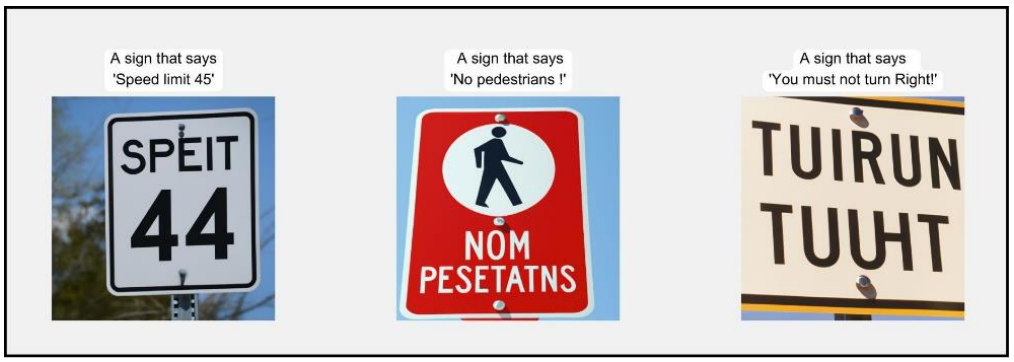


Fig. 6. Text limitations of the DALL-E 2 for (512,512).

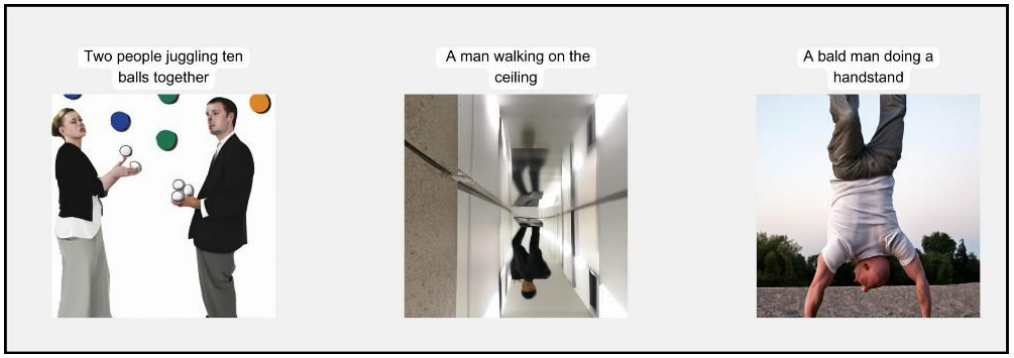


Fig. 7. Limitations concerning people's faces in the DALL-E 2 for (512,512).

Fig. 8. Text limitations of the Pix-Art-α for (512,512).

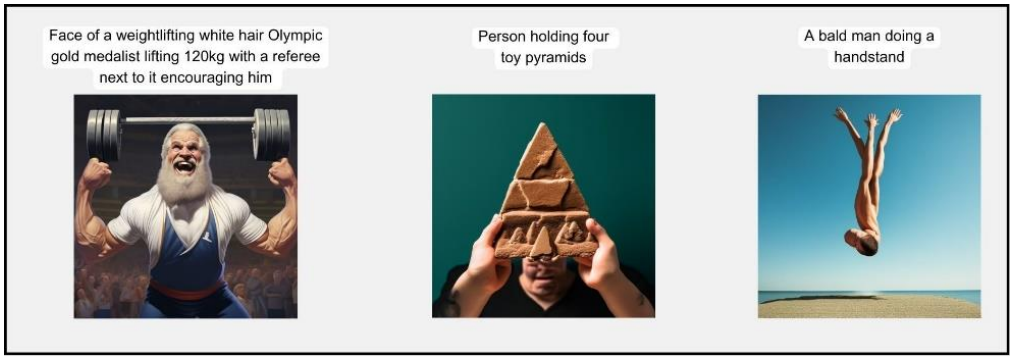


Fig. 9. Limitations in rendering fingers and hands in the Pix-Art-α for (512,512).

3 CONCLUSION

This study evaluates three text-to-image models using both human assessments and the CLIP score metric, providing insights into model behavior based on the applied hyperparameters.

In the human evaluation, DALL-E 2 consistently outperforms the open-source models for 512x512 image resolution, particularly excelling in coloring, counting, facial features, and text rendering. However, it struggles with facial accuracy when faces are not explicitly mentioned in the prompt, often misrepresenting eye pupils. At 1024x1024 image resolution, DALL-E 2 maintains its lead across all tasks. Pix-Art-α ranks second at 512x512 image resolution, performing well in conflicting and positional tasks, but at 1024x1024 image resolution, its performance declines, excelling only in conflicting tasks. Stable Diffusion performs poorly across all six tasks at both 512x512 and 1024x1024 image resolution, demonstrating weak prompt alignment.

In the CLIP score evaluation, DALL-E 2 also achieves the best text-image alignment for 512x512 image resolution, leading in coloring, facial features, and text tasks. Pix-Art-α ranks second, performing well in conflicting and positional tasks, while Stable Diffusion only shows strength in counting tasks. For 1024x1024 image resolution, DALL-E 2 retains the top position, excelling in conflicting, counting, positional, and text tasks. Pix-Art-α remains second, with strong performance in coloring and facial features, while Stable Diffusion continues to underperform across all tasks.

REFERENCES

[1] Kappal, S.: Data Normalization using Median and Median Absolute Deviation (MMAD) based Z-Score for Robust Predictions vs. Min-Max Normalization London Journal Press 19(4) (2019)

[2] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., et al.: Learning Transferable Visual Models From Natural Language Supervision. (2021). doi: 10.48550/arXiv.2103.00020

[3] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical Text-Conditional Image Generation with CLIP Latents. (2022). doi: 10.48550/arXiv.2204.06125

[4] Openai API documentation. https://platform.openai.com/docs/api-reference

[5] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-Resolution Image Synthesis with Latent Diffusion Models. (2022). doi: 10.48550/arXiv.2112.10752

[6] Rombach, R., Esser, P.: Huffing face, stable-diffusion-v1-5 model card. (2023) https: //huggingface.co/stable-diffusion-v1-5/stable-diffusion-v1-5

[7] Chen, J., Yu, J., Ge, C., Yao, L., Xie, E., Wu, Y., et al.: PixArt-α: Fast Training of Diffusion Transformer for Photorealistic Text-to-Image Synthesis. (2023). doi: 10.48550/arXiv.2310.00426

[8] Pixart-alpha model card. https://huggingface.co/PixArt-alpha/PixArt-XL-2-1024-MS

[9] Pixart-alpha 1024px. https://huggingface.co/spaces/PixArt-alpha/PixArt-alpha

[10] Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J.: Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. (2022). doi: 10.48550/arXiv.2205.11487

[11] Petsiuk, V., Siemenn, A. E., Surbehera, S., Chin, Z.: Human evaluation of text-to-image models on a multi-task benchmark. (2022). doi: 10.48550/arXiv.2211.12112

[12] Cahyadi, M., Rafi, M., Shan, W., Moniaga, J., Lucky, H.: Accuracy and Fidelity Comparison of Luna and DALL-E 2 Diffusion-Based Image Generation Systems. (2023). doi: 10.48550/arXiv.2301.01914

[13] OpenAI CLIP model. https://github.com/openai/CLIP.