Deep Learning Course – Final Project

Leveraging Deep Learning Models for Salary Prediction

Aviel Edri 208244012 Itamar Kraitman 208925578 Yuval Shabat 318516630

Abstract

In this study, we explore the application of deep learning models for predicting salaries based on various job-related features. We investigate the performance of different deep learning architectures, including linear regression, Multilayer Perceptron (MLP), MLP with adjusted weights, and Convolutional Neural Network (CNN), on a dataset containing information about job titles, experience levels, employment types, and other relevant attributes. Through extensive experiments and evaluations, we aim to identify the most effective model for salary prediction tasks. Leveraging a dataset comprising diverse job, the models are trained and evaluated to ascertain their predictive capabilities.

Introduction

The prediction of salary is a crucial task in various industries, aiding in hiring decisions, compensation planning, and workforce management. Traditional methods of salary prediction often rely on simplistic regression models or statistical techniques, which may not capture the complex relationships between job-related features and salary accurately. With the advancements in deep learning, there has been growing interest in leveraging neural networks to tackle salary prediction tasks. In this paper, we present a comprehensive analysis of deep learning models for salary prediction, aiming to provide insights into their effectiveness and practical implications.

Related Work and Required Background

Previous research in salary prediction has primarily focused on conventional machine learning techniques, such as linear regression, decision trees, and random forests. While these methods have shown some success, they often struggle to handle nonlinear relationships and high-dimensional data effectively. Deep learning models offer a promising alternative, capable of automatically learning intricate patterns and representations from raw data. Convolutional Neural Networks (CNNs) and Multilayer Perceptrons (MLPs) have emerged as popular choices for various regression tasks, including salary prediction. Additionally, familiarity with concepts like neural networks, activation functions, and loss functions is essential for understanding the experiments conducted in this study.

Project Description

Dataset Overview

The dataset used in this project contains comprehensive information related to job positions and corresponding salaries. It includes attributes such as work year, experience level, employment type, job title, salary, salary currency, salary in USD, employee residence, remote work ratio, company location, and company size. This dataset provides a rich source of features that serve as indicators for determining the annual salary (in USD), allowing for a detailed exploration and analysis.

	work_year	experience_level	employment_type	job_title	salary	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2023	SE	FT	Principal Data Scientist	80000	EUR	85847	ES	100	ES	L
1	2023	MI	CT	ML Engineer	30000	USD	30000	US	100	US	S
2	2023	MI	CT	ML Engineer	25500	USD	25500	US	100	US	S
3	2023	SE	FT	Data Scientist	175000	USD	175000	CA	100	CA	M
4	2023	SE	FT	Data Scientist	120000	USD	120000	CA	100	CA	M

As shown, the dataset composed out of the following features:

- **Work Year:** The calendar year in which the salary was disbursed.
- **Experience Level:** The professional experience tier of the individual in their respective role for the given year. There are 4 experience levels, EN for Entry Level, MI for Mid-level, SE for Senior-level, and EX for Executive-level.
- **Employment Type:** The nature of the employment contract for the specified position. There are also 4 employment types, CT for Contract, FL for Freelance, FT for Full Time, and PT for Part Time.
- **Job Title:** The designation or title held by the employee during the said year.
- **Salary:** The aggregate gross salary amount remunerated.
- **Salary Currency:** The currency in which the salary was paid, codified using the ISO 4217 standard.
- **Salary in USD:** The equivalent salary value converted to United States Dollars.
- **Employee Residence:** The primary country of residence of the employee, encoded as per the ISO 3166 standard.
- **Remote Ratio:** The proportion of work conducted remotely, represented as a percentage.
- **Company Location:** The country where the employer's main office or principal contracting branch is situated.
- **Company Size:** An estimate of the median workforce size within the company throughout the year.

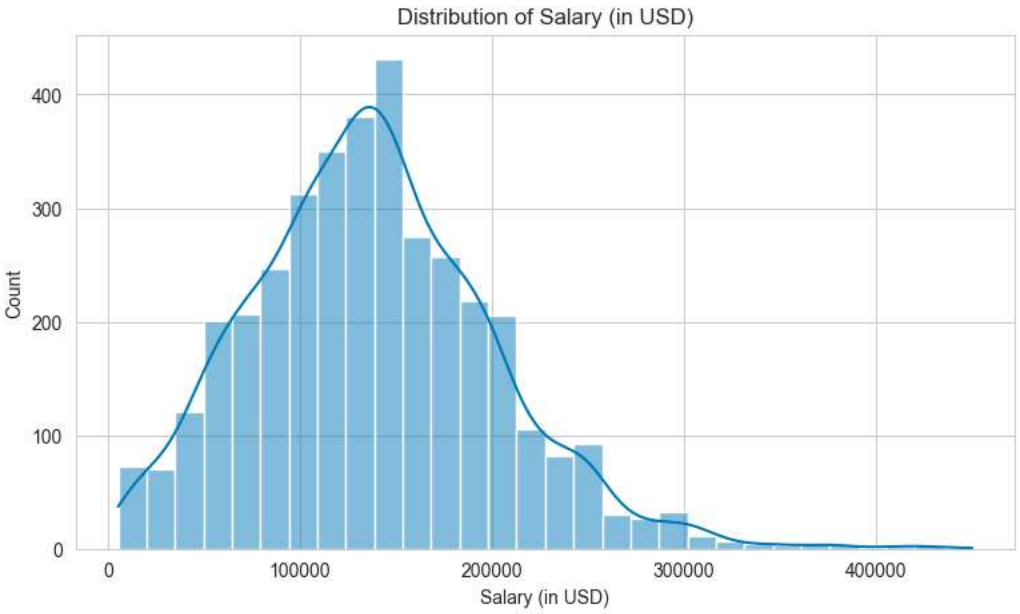
Analysis

Before constructing predictive models, we conducted an in-depth analysis of the dataset to gain insights into the distribution and relationships among different features. When dealing with a dataset of such size, it is important to verify there are no redundant data, no empty or Nan cells, and all the given data is effective for the model's training.

This analysis involved visualizing various aspects of the data through relevant plots and charts. Some of the plots created during this phase include:

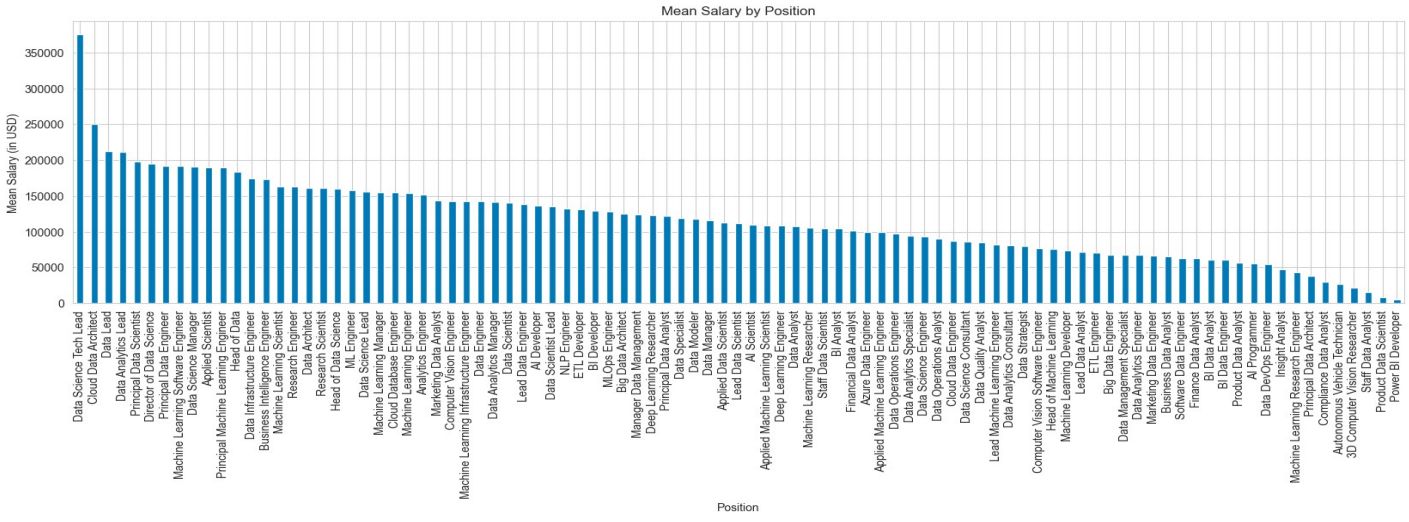
Salary in USD Distribution

Visualizing the distribution of salary (in USD) values to understand the range and spread of salaries across the dataset.



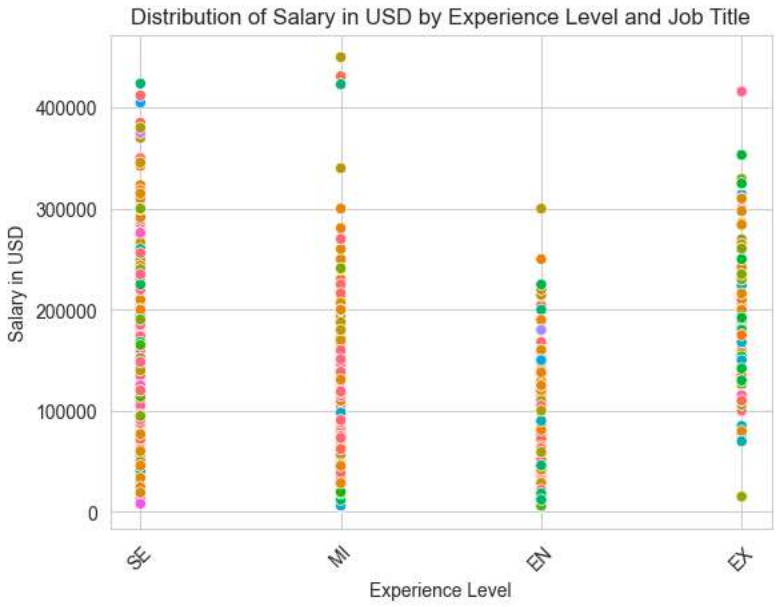
Mean Salary in USD by Position

Investigating the relationship between job titles and salary levels to identify any significant variations or outliers.



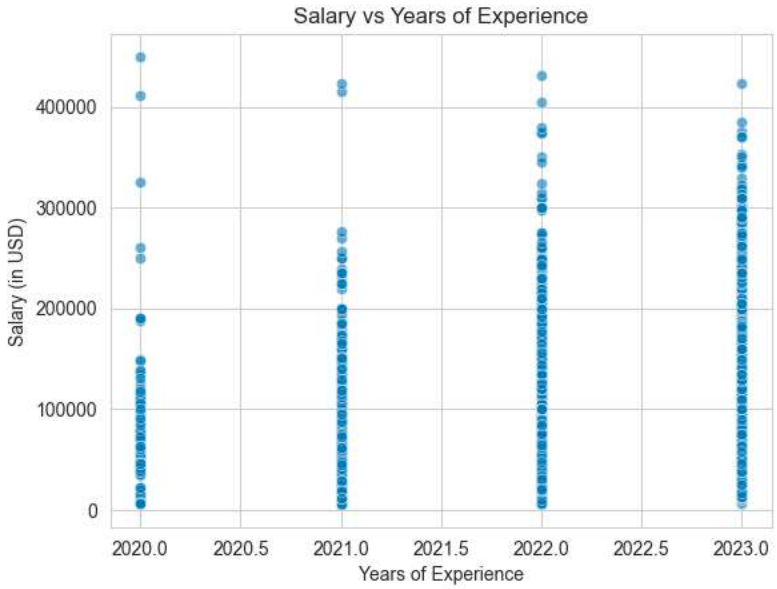
Distribution of Salary in USD by Experience Level and Job Title

Visualizing the distribution of salary (in USD) values to understand the range and spread of salaries across the dataset by experience level and job title.



Salary in USD Distribution by Years of Experience

Visualizing the distribution of salary (in USD) values to understand the range and spread of salaries across the dataset by the years of experience.



Data Normalization and *Preprocessing*

The structure of the data used for training and validation is crucial for the effectiveness and generalization ability of the models. Therefore, we preformed several modifications to the dataset to allow the models to understand and process these features effectively.

Normalization for Numerical Features

Normalizing numerical features ensures that they are on a similar scale, preventing features with larger magnitudes from dominating the learning process. This step helps improve the stability and convergence of optimization algorithms during model training, especially for the models that are sensitive to feature scales. The current dataset salary columns (salary_in_usd, salary), represented by large numbers. Therefore, we optimize the numbers and divide the features by 100,000. Then, we utilized *StandardScaler* from the *scikit-learn* library to normalize the data, ensuring uniform scaling.

Handling Categorical Variables

The dataset contains categorical variables, such as job titles or employment types, which cannot be directly used as input for the models. Encoding categorical variables into numerical representations, allows the models to understand and process these features effectively. Categorical features were subjected to one-hot encoding using *OneHotEncoder* from the *scikit-learn* library. This step aimed to transform categorical data into a format suitable for numerical processing while accommodating potential unknown categories during inference.

Preprocessing

The preprocessing steps were orchestrated into a cohesive pipeline using `ColumnTransformer` to facilitate simultaneous transformation of both numerical and categorical features.

Convolutional Neural Network (CNN) Model

Model Structure

The CNN model consists of multiple layers designed to extract hierarchical features from the input data. Here is the detailed structure of the CNN model:

- Convolutional Layers: Three convolutional layers are used to capture spatial patterns in the input data. Each convolutional layer is followed by a Rectified Linear Unit (ReLU) activation function to introduce non-linearity.
  - The first convolutional layer has 32 filters with a kernel size of 3 and takes an input shape of *X_train_preprocessed* set.
  - The subsequent convolutional layers have 64 and 128 filters, respectively, with the same kernel size of 3.
  - MaxPooling layers with a pool size of 2 are added after each convolutional layer to downsample the feature maps.
- Flatten Layer: After the convolutional layers, a Flatten layer is applied to convert the 3D feature maps into a 1D vector, preparing the data for the fully connected layers.
- Dense Layers: Two fully connected Dense layers are added to the model to perform the final classification. The first Dense layer has 64 units with a ReLU activation function, followed by a single output unit without an activation function.

Optimization

The model is compiled using the Adam optimizer. The key parameters of the Adam optimizer include the learning rate (set to 0.0001 in this model), which controls the step size during optimization.

Training

During training, the model minimizes the mean squared error (MSE) loss function, which measures the average squared difference between the predicted and actual salary values. Additionally, Mean Absolute Error (MAE) and Mean Squared Error (MSE) are used as evaluation metrics to assess the model's performance on the validation set.

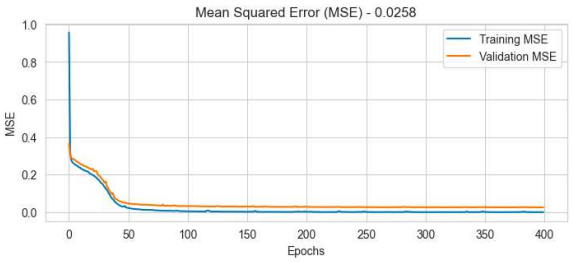
The model is trained for 400 epochs with a batch size of 32. The training and validation datasets are used to update the model's parameters and evaluate its performance iteratively.

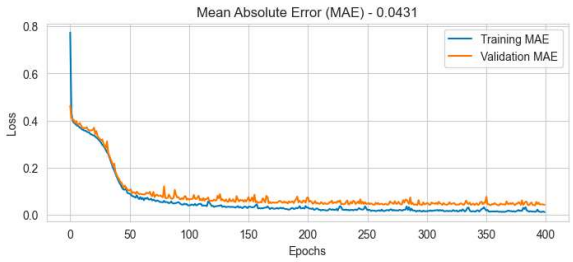
Prediction

Once trained, the CNN model is used to predict salary values for the validation dataset (X_valid_preprocessed). The model's predictions are obtained using the cnn_model.predict() function, which generates predicted salary values based on the preprocessed input data.

Experiments Results

The experiments conducted in our study reveal interesting findings regarding the performance of deep learning models for salary prediction. The CNN model demonstrates superior performance:

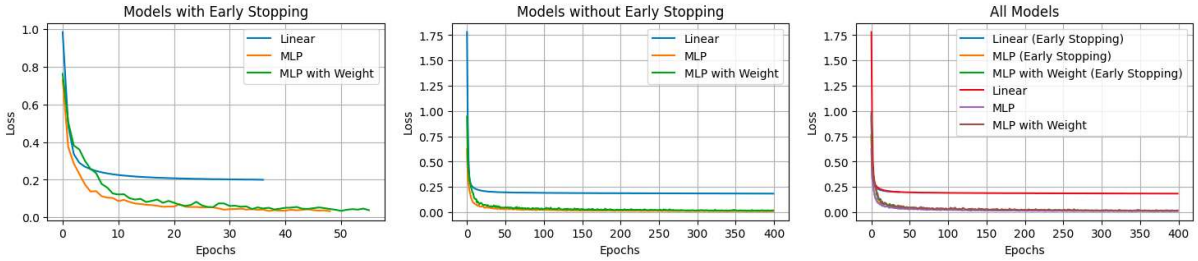




Where in the plot below shows the predication performance scattered:

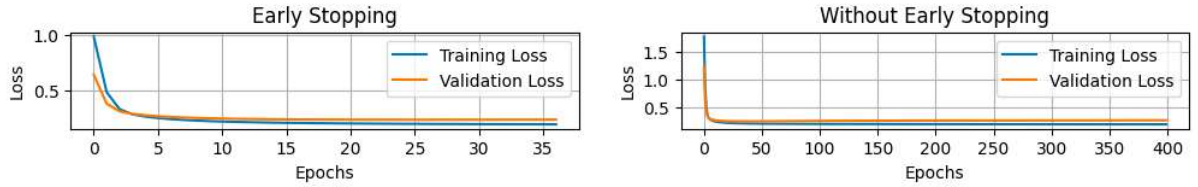
Previous Attempts

In our preliminary endeavors to construct predictive models for estimating salaries of Data Science positions worldwide, we adopted various methodologies and explored multiple machine learning architectures. We experiment with linear regression, Multilayer Perceptron (MLP), MLP with adjusted weights, and Convolutional Neural Network (CNN) models, tuning hyperparameters and employing early stopping techniques to prevent overfitting. We evaluate the performance of each model using metrics like Mean Absolute Error (MAE) and Mean Squared Error (MSE). Throughout these attempts, common issues emerged, including overfitting, premature convergence, and limited gains in predictive accuracy despite the complexity of the model architectures.



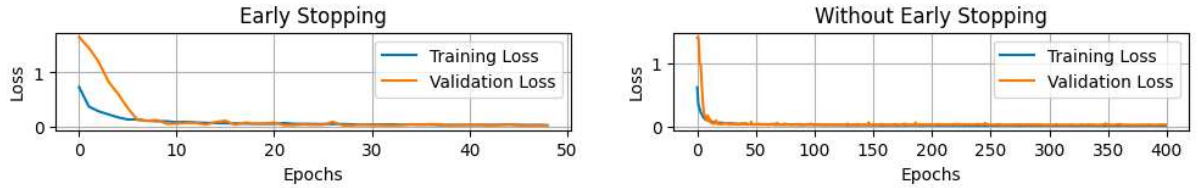
Linear Regressing Models

The initial approach focused on linear regression models, which struggled to capture the intricate relationships present in the dataset due to their inherent simplicity and linearity assumptions. Despite preprocessing techniques such as feature scaling and one-hot encoding to prepare the data, these models exhibited limited predictive accuracy. The mean squared error (MSE) on the validation set remained relatively high, indicating suboptimal model performance.



Multi-Layer Perceptron (MLP) Models

Subsequent attempts involved the exploration of more sophisticated neural network architectures, including MLPs, to address the limitations of linear models. While MLPs offer greater flexibility in capturing non-linear relationships through multiple hidden layers, they faced challenges such as overfitting and premature convergence. Despite incorporating dropout regularization and batch normalization techniques, the MLP models struggled to generalize well to unseen data, leading to limited improvements in predictive accuracy.



Conclusions

In conclusion, our study highlights the potential of deep learning models, particularly CNNs, in salary prediction tasks. The experiments demonstrate that CNNs can effectively leverage the spatial information embedded in the input data, leading to improved performance compared to traditional MLP models. While each attempt provided valuable insights into the intricacies of the task, including the importance of feature engineering and regularization techniques, none fully addressed the challenges like the CNN model.