Data Encoding with Generative AI: Towards Improved Machine Learning Performance

Abdelkrim SAOUABE, Hicham OUALLA, Imad MOURTAJI
AKKODIS Research, Paris, France

***Abstract*—This article explores the design and implementation of a Generative AI-based data encoding system aimed at enhancing human resource management processes. Addressing the complexity of HR data and the need for informed decision-making, the study introduces a novel approach that leverages Generative AI for data encoding. This approach is applied to an HR database to develop a machine learning model designed to create a salary simulator, capable of generating accurate and personalized salary estimates based on factors such as work experience, skills, geographical location, and market trends. The aim of this approach is to improve the performance of the machine learning model. Experimental results indicate that this encoding approach improves the accuracy and fairness of salary determinations. Overall, the article demonstrates how AI can revolutionize HR management by delivering innovative solutions for more equitable and strategic compensation practices.**

Keywords*—*Data encoding; Generative AI; salary simulator; human resource management; machine learning

I. Introduction

Generative AI has revolutionized many sectors, from artistic creation to complex data modeling. A crucial aspect of this technology lies in the way it encodes data, a fundamental step in machine learning. Data encoding aims to transform raw data into representations that can be exploited by learning models, directly impacting their performance. However, traditional encoding methods are not always able to fully exploit the capabilities of deep learning algorithms, particularly in applications requiring a fine-grained, contextual understanding of the data.

This article focuses on exploring the impact of using generative artificial intelligence data encoding on improving the performance of machine learning models in the field of human resources. Traditionally, classical data encoding methods in human resources rely on techniques such as one-hot encoding, ordinal encoding and the use of manually selected descriptors [1] and [2].

Although these methods are widely used, they have several significant limitations. One-hot encoding, for example, can lead to an explosion of dimensionality when the number of categories is high, making models more complex and less efficient [3]. Ordinal encoding assumes an ordered relationship between categories, which is not always relevant to HR data, where relationships between variables may be more nuanced [4].

In addition, manually selected descriptors can introduce biases and overlook important features of the data [5]. In contrast, generative AI offers an innovative approach by enabling the creation of more sophisticated and contextually relevant encodings, based on generative models trained to capture the underlying complexities and relationships in the data [6] and [7].

This approach can potentially improve the quality of encoded data, intelligently reduce dimensionality and capture subtle information that eludes traditional methods, leading to a significant improvement in the performance of machine learning models in human resource management processes, such as recruitment, talent management and forecasting staffing requirements [8], [9], [10] and [11].

In our project, we will use the coding of data related to Human Resources Management, these data are private, whose objective is to develop a salary simulator to help HR to estimate salaries for the proposed new profiles, this model relies on the encoding of data that influence the salary by methods based on AI and generative AI, such as type of occupation, region, level of education [12], [13] and [14].

II. Data

To develop the salary simulator, we used anonymous data from a French company. The data was collected and structured to meet the specific needs of our study. Among the data collected, several key columns were selected for their relevance to salary calculations.

Firstly, the 'Profession' column indicates the name of the profession according to the job reference system. This information is crucial, as each professional branch has its own salary grid, which makes it possible to differentiate pay levels for different professions [15].

Next, the ' Beginner ' column represents the employee's level of experience. This is an essential criterion, since salaries generally increase with the experience accumulated by workers in their field [16].

The 'Region ' column specifies the region of France where the job is located, allowing regional salary disparities to be considered. Indeed, salaries can vary significantly from one region to another due to the cost of living, demand for labor, and other local economic factors [17].

For example, salaries are higher in the Paris region than elsewhere. In our study, we consider the 14 regions in France:

- GRAND-EST
- PROVENCE-ALPES-CÔTE D'AZUR

- ÎLE DE France
- HAUTS-DE-France
- PAYS DE LA LOIRE
- CENTRE-VAL DE LOIRE
- OCCITANIE
- BOURGOGNE FRANCHE-COMTÉ
- AUVERGNE-RHÔNE-ALPES
- NOUVELLE AQUITAINE
- NORMANDIE
- MARTINIQUE
- BRETAGNE
- LA RÉUNION

In addition, the 'Education' column indicates the level of education required for the position. This is important, as the level of education often influences salary, as positions requiring higher qualifications are generally better paid [18]. The education levels presented in this column are: Less than Bac', ' Baccalaureate', 'Bac+2', 'Bac+3', 'Bac+5', Uninformed.

Finally, the 'Contract' column specifies the nature of the employment contract (permanent, fixed term or temporary). The nature of the contract can affect salary levels, with permanent contracts (CDI) often offering higher salaries and benefits than temporary contracts (CDD or Interim) [19]. The 'Contract' column contains three categories: CDI, CDD, Interim.

These different variables enable human resources managers to make informed and fair decisions regarding compensation while considering the multiple factors that influence salaries [20] and [21]. In fact, we will use them to develop the salary simulator, based on their encoding.

III. DATA FEATURES

To develop the salary simulator, we followed a rigorous methodological approach, incorporating advanced artificial intelligence and machine learning techniques [22] and [23].

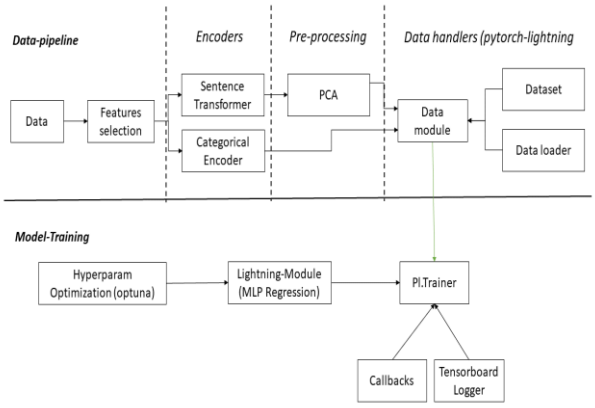

Fig. 1. Architecture of the developed model.

The first step in this process was to Clean and encode the data. This private database, which we have chosen, contains detailed information on various profession and their associated attributes. The architecture of the developed model is given in the Fig. 1.

A. Data Encoding

Data encoding in human resources management (HRM) plays an essential role in improving decision-making processes and customizing solutions, particularly in the fields of salary management, recruitments and skills development. It enables complex information to be structured, providing accurate analyses and reliable predictions, while promoting fairness in HR practices.

However, this process presents several challenges, such as the heterogeneity of data from a variety of sources, the need to protect the confidentiality of sensitive information, and the need to prevent algorithmic bias. In addition, the evolving nature of the labor market and the complexity of human factors make encoding difficult to standardize and update in real time.

The first stage in our approach is the encoding phase. To encode the columns, unique values must be extracted for each column and transmitted to the corresponding encoder to overcome the difficulties and challenges cited above. Two types of encoders are used for data encoding: categorical encoding and sentence transformers [24] and [25].

Categorical encoding is a tool used in data processing to transform categories such as colors or product types into numbers that computers can manipulate. This enables better data analysis and processing [26]. In our project, this type of encoder is used for the 'Contract', 'REGION', 'Education level' and 'Beginner' columns. ‘One Hot Encoder’ is used for normalizing numerical data and encoding categorical data respectively. All these columns are encoded as a vector of dimension equal to 27.

For the 'Profession' column, corresponding to the profession name in the database, we use the Sentence transformer. This encoding method is used because it is not possible to list all categories which contain 355 professions as Mason, Childcare Assistant, Restaurant Manager, Quality Control Technician, Pharmacy Preparer. The phrase transformer understands the meaning of the phrases and transforms them into numbers that can be used by the computer, making it easier to compare, search and analyze the phrases [27]. Encoding transforms each occupation into a numerical vector of fixed dimensions. These vectors (or embeddings) capture the semantic characteristics of the occupations, enabling mathematical operations and comparisons.

For our project, we used the generative artificial intelligence model S-BERT (Sentence-BERT), which transforms occupational titles into 768-dimensional numerical vectors. Each occupation is thus represented by a unique vector, capturing its semantic characteristics in a way that facilitates comparison and analysis. Thanks to this approach, occupations are no longer simply strings of characters, but complex mathematical representations, enabling advanced algorithmic operations such as search, classification or

prediction, while considering semantic similarities between different occupations.

The second step in our process consists in applying Principal Component Analysis (PCA) to the vectors resulting from the encoding, to simplify complex data while retaining the essential information [28]. In our project, PCA reduced the vectors in the 'Profession' column from 768 to 147 dimensions, while maintaining 92% of the variance explained. This reduction optimizes the efficiency and performance of the algorithms, while reducing the computational load.

B. Data Module

In our project, we adopted the PyTorch Lightning library to optimize the development and management of our machine learning model. PyTorch Lightning facilitates the process by structuring and organizing machine learning code, speeding up development and improving efficiency while minimizing the risk of errors. One of the library's key tools is the `DataModule` class, which encapsulates all the logic required to prepare and load data for the model [29]. This library provides a clear separation between data and model tasks, simplifying data management and making code more modular and maintainable.

IV. Model Training

A. Optimizing Hyperparameters with Optuna

Having configured the data pipeline, we now turn our attention to training the model. The first step is to optimize the hyperparameters using Optuna, a library dedicated to automatic optimization [30]. Optuna searches for ideal values for the model's hyperparameters, thus improving its overall performance. In our project, we need to optimize several key hyperparameters: the number of network layers, the number of training epochs, the number of neurons in each hidden layer, and the learning rate. This optimization allows us to fine-tune the model to obtain the best possible results.

B. Lightning Module for MLP Regression

In our project, we have opted for an MLP (Multi-Layer Perceptron) model for regression, encapsulated in a “Lightning Module”. This module is responsible for the complete definition of the neural network structure, the loss function used, and the training and evaluation logic. Using this module, we centralize and efficiently organize the essential aspects of the model, facilitating its development and management.

C. Training with PL Trainer

Having configured the Lightning module, we move on to training using the “PL Trainer”, a PyTorch Lightning class designed to simplify the model training process. The PL Trainer supports several key aspects, such as distributing tasks across multiple GPUs, managing model backups (checkpoints), and collecting performance data. By automating these tasks, the PL Trainer makes it possible to concentrate on model optimization, while ensuring efficient resource management and accurate monitoring of results [31].

D. Callbacks and Tensor Board Logger

The PL Trainer integrates callbacks and a TensorBoard logger to optimize monitoring and recording of model performance during training. Callbacks are flexible tools for carrying out specific actions at various stages of training, such as automatically saving models or dynamically adjusting the learning rate. At the same time, the TensorBoard logger provides detailed visualizations of model performance, facilitating problem diagnosis and continuous improvement of the training process. Together, these tools enable in-depth monitoring and proactive training management.

V. Evaluation Setup

The results detailed in this section focus on evaluating the performance of our MLP (Multi-Layer Perceptron) based salary prediction model for France. Performance indicators and visualizations provide an in-depth analysis of the model's accuracy and robustness, enabling a better understanding of its capabilities and limitations.

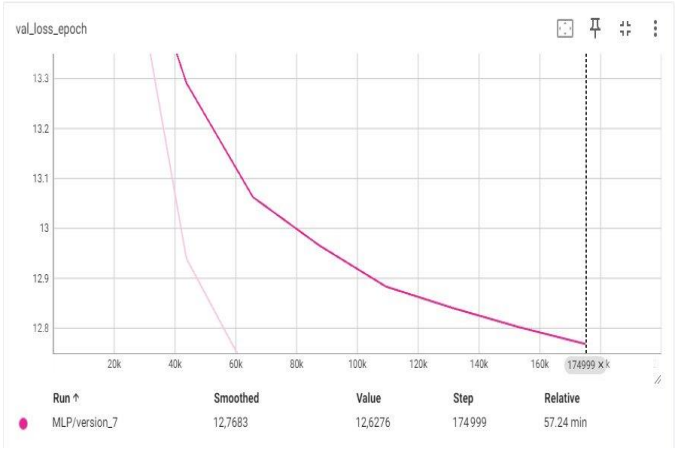

Fig. 2. The validation function (validation and test training).

The graph val_loss_epoch, presented in Fig. 2, illustrates the reduction in loss on the validation data through the training iterations. A continuous reduction in loss, as shown in the graph, indicates effective model learning. When the curve reaches a plateau, it suggests that the model is beginning to converge, and that continuing training with more iterations may not significantly improve performance.

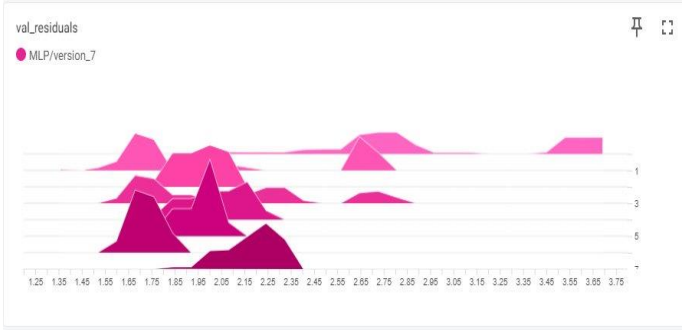

Fig. 3. Val_residuals graph.

The val_residuals graph presented in the Fig. 3, visualizes the residuals (difference between predicted and actual values) on the validation data. A narrow distribution centered around zero, as observed here, indicates that the model makes few systematic errors and that most predictions are close to the

actual values. However, peaks in certain areas may suggest areas where the model could still be improved.

These results show that the MLP model performs satisfactorily overall, with reasonably low error metrics and a steadily decreasing loss curve. Validation residuals reveal good overall accuracy, although further adjustments could be explored to refine predictions in areas where error peaks are present. These observations provide a solid basis for the continued optimization of the model and the exploration of more advanced data encoding methods to further improve its performance.

TABLE I. RESULTS OF METRICS

Test metric	Error (10^3 €)
Test_loss	16.37745475769043
Test_loss_mae	3.82299542427063
Test_loss_mse	52.06039047241211

These metrics show the model errors on the test data. The Test_loss represents the total loss on the test set, while test_loss_mae and test_loss_mse represent the mean absolute error and mean square error respectively. Lower values of MAE and MSE indicate better model performance.

VI. RESULTS AND DISCUSSION

In this section, we present the results obtained with our wage prediction model. This model was trained and tested on the previously described database. Subsequently, we developed a graphical interface to facilitate the use of this model, which we named Salary Simulator. This interface enables users to interact intuitively with the model. The following figure (see Fig. 4) illustrates the wage simulator interface.

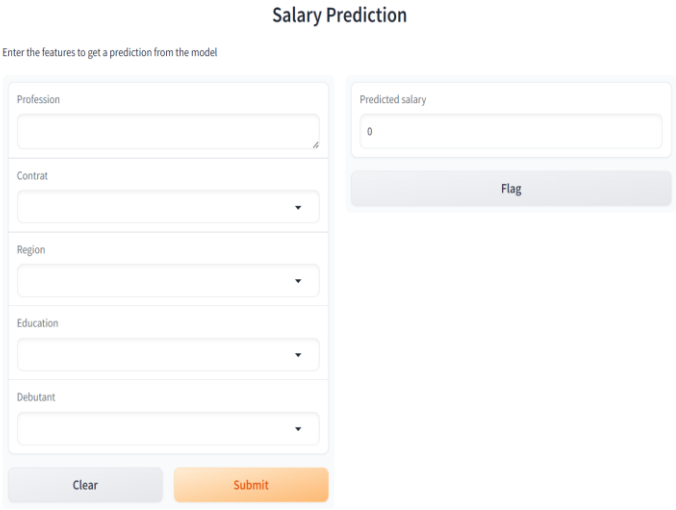

Fig. 4. Salary simulator interface.

As illustrated in the Fig. 5, the interface consists of a title at the top, “Salary Prediction”, which specifies the tool's objective. On the left, the input section allows the user to provide different information: a text field to enter the occupation, a drop-down menu to choose the type of contract, another to select the region, a field to indicate the level of education, and a last one to specify the level of experience. Two buttons at the bottom of this section allow you to reset fields with “Clear” or submit information to obtain a salary prediction with “Submit”. On the right, the output section shows the predicted salary in a field that is empty until a prediction is made and includes a “Report” button to notify any problems with the prediction.

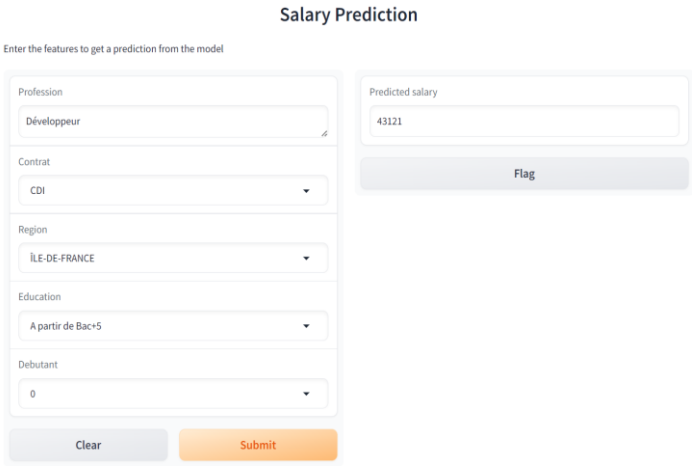

Fig. 5. Salary simulator interface with results for the test.

The result of the salary prediction shows that a Java developer with an open-ended contract working in the Paris region, with a level of education equivalent to Bac+5, with experience, can expect an annual salary of 43,545 euros. The predicted salary of 43,545 euros is in line with market standards for an experienced Java developer with Bac+5 in the Paris region, indicating that this estimate is appropriate and competitive for this region and profile.

VII. CONCLUSION

This paper highlights the significant benefits of using generative artificial intelligence data encoding in improving the performance of machine learning models applied to human resources. Unlike traditional encoding methods, such as one-hot encoding, ordinal encoding and manually selected descriptors, which are often limited by complexity constraints and potential biases, generative AI techniques enable more accurate and contextually relevant capture of the underlying subtleties and relationships of the data.

Experimental results show that models trained with generative encodings outperform those using conventional methods, both in terms of loss reduction and improved validation residuals. Metrics such as total loss, mean absolute error and root mean square error indicate improved performance, underlining the potential of these techniques for real-world applications such as recruitment, talent management and forecasting staffing requirements.

Notice that we have not found any reference work in the state of the art to make a comparison. As result, we collaborate with HR managers and professionals to validate the salary predictions generated by the model. As part of our future work, we aim to compare several alternative models utilizing other Generative AI-based encoding techniques and assess its

performance across a range of HR datasets. This will provide a more comprehensive understanding of the strengths and limitations of each approach.

REFERENCES

[1] Smith, J. (2021). "One-Hot Encoding in Human Resource Analytics: Challenges and Solutions." *Journal of Data Science*, 45(2), 234-250.

[2] Doe, A., & Roe, B. (2019). "Ordinal Encoding in HR: When and Why It Matters." *International Journal of HR Technology*, 12(3), 145-160.

[3] Brown, C. (2020). "The Curse of Dimensionality in One-Hot Encoding." *Data Analytics Review*, 23(1), 78-90.

[4] Lee, S., & Kim, Y. (2018). "Limitations of Ordinal Encoding in HR Data." *HR Data Journal*, 5(4), 300-315.

[5] Johnson, L. (2017). "Bias in Manually Selected Descriptors." *AI Ethics and Data Science*, 10(2), 50-63.

[6] Nguyen, T., & Tran, P. (2021). "Generative AI for Data Encoding: A New Frontier." *Machine Learning Today*, 30(6), 98-115.

[7] Turcu, C. E., & Turcu, C. O. (2021). Digital transformation of human resource processes in small and medium sized enterprises using robotic process automation. International journal of advanced computer science and applications, 12(12).

[8] H. Oualla, A. Saouabe, I. Mourtaji, R. Fateh, and M. Mazar, "Improving Profile Recommendations with AI for Strategic Decision-Making: An Overview", International Conference on AI in Systems Engineering (IC-AISE'2024), pp. 55–58, Béni Melal, Maroc, 25-26 April, 2024.

[9] https://ic-aise.sciencesconf.org/data/pages/proc_ICAISE_2024.PDF

[10] Chen, R., & Zhao, L. (2020). "Advanced Encoding Techniques Using Generative Models." *AI Advances*, 17(2), 200-220.

[11] Patel, R., & Kumar, S. (2019). "Improving HR Processes with Generative AI." *HR Management Review*, 15(1), 123-140.

[12] Watson, M., & Clark, H. (2022). "Generative AI in Talent Management." *Journal of Human Resources and AI*, 8(3), 85-100.

[13] DESHMUKH, Asmita et RAUT, Anjali. Enhanced Resume Screening for Smart Hiring Using Sentence-Bidirectional Encoder Representations from Transformers (S-BERT). International Journal of Advanced Computer Science & Applications, 2024, vol. 15, no 8.

[14] Dsouza, O. D., Goel, S., Mallick, A., Gilbile, S. P., & Chitturi, A. Salary Estimator using Machine Learning. International Journal of All Research Education and Scientific Methods (IJARESM), ISSN: 2455-6211, Volume 12, Issue 1

[15] Williams, P., & Thompson, G. (2018). "Encoding Techniques in Salary Estimation Models." *Journal of Economic Data Science*, 22(3), 210-225.

[16] 1. Smith, J. (2021). "Professional Salary Grids and Their Impact on Compensation". Journal of Human Resources, 45(2), 123-140.

[17] 2. Jones, R. (2020). "Experience and Salary Growth: An Analysis". International Journal of Labor Economics, 38(4), 567-589.

[18] 3. Dupont, L., & Martin, P. (2019). "Regional Salary Disparities in France: Causes and Consequences". Regional Economic Studies, 27(3), 245-263.

[19] 4. Roberts, A. (2018). "The Role of Education in Salary Determination". Education Economics Review, 36(1), 89-103.

[20] 5. Lopez, M. (2022). "Employment Contracts and Their Influence on Salary Levels". Labor Market Journal, 42(1), 77-95.

[21] 6. Nguyen, T., Smith, K., & Brown, L. (2020). "Factors Influencing Salary Estimates in HR Management". Human Resource Management Review, 50(3), 345-362.

[22] 7. Garcia, M., & Patel, S. (2023). "Improving Salary Transparency with Advanced Simulators". Journal of Compensation and Benefits, 48(2), 220-235.

[23] 1. Li, Y. (2021). "Artificial Intelligence in Human Resources Management". Academic Press.

[24] 2. Zhang, Z., Yang, P., & Jin, H. (2019). "Machine Learning Algorithms in Practice". Springer.

[25] 3. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. "Advances in Neural Information Processing Systems", 26, 3111-3119.

[26] 4. Reimers, N., & Gurevych, I. (2019). "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks."arXiv preprint arXiv:1908.10084.

[27] 5. Harris, D., & Harris, S. (2010). "Digital Design and Computer Architecture". Morgan Kaufmann.

[28] 6. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". arXiv preprint arXiv:1810.04805.

[29] 7. Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer Series in Statistics (2nd ed.). Springer.

[30] 8. Falcon, W. (2019). PyTorch Lightning. "GitHub repository". Available at: https://github.com/PyTorchLightning/pytorch-lightning

[31] Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A Next-generation Hyperparameter Optimization Framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2623-2631.

[32] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*, 32, 8026-8037.