Apply Machine Learning to Disambiguate Inventors Using Images in Patents

Sheldon Li, Hao Yang, Zengyu Yang, Guillaume Domenge

February 9, 2025

1 Executive Summary

The ability to differentiate between the inventors of patents plays an important role in technology management and breakthrough prediction. Our capstone project aims to utilize machine learning and computer vision to help disambiguate inventors of patents. In this section, we will introduce the significant development path and research in this field.

1.1 Significance of patent differentiation for the industry and research

Inventors are identified as a critical factor in making technological breakthroughs. Knowing the career path of the inventors could benefit not only research into invention policy, but also the innovation strategy for tech companies. In 2003, Vetle I. Torvik [1] presented a model for automatically generating training sets and estimating the probability that a pair of Medline records, the database for the National Library of Medicine, with a last and first name initial have the same inventor, and he found that pairs of different articles inventored by the same individuals do share similarities. The most powerful feature for distinguishing is the number of common co-inventors, followed by journal match, and then middle initial match. In 2009, Torvik and Smalheiser [2] named the model "inventor-ity", which is used to estimate the probability that two articles in Medline, sharing the same inventor name, were written by the same inventor. inventor name disambiguation allows information retrieval and data integration to become person-centered, not just document-centered, opening a new stage for new data management and social network tools that will facilitate the analysis of scholarly publishing and collaboration behavior.

1.2 Identifying the problems in the patent disambiguation scenario

With the new disambiguation method, there are still challenges to uniquely identify inventors. The United States Patent and Trademark Office does not require consistent and unique identifiers for inventors, which makes it difficult to ensure that the two patents have exactly the same inventor. Besides, a big part of the databases lost the text data, which means that the database has no text-based features that are helpful to differentiate the inventors.

1.3 Innovatively including images of the patent as our features

With the existing patent database, we have limited text data from each patents. However, with the access to images data from each patent, we choose to include the images in the patent as our features and build new databases with images features by calculating similarity scores between images from different patents in order to predict whether these two patents have the same inventors or not. In this way, we could avoid the problem of not having enough text data of patents.

The focus of our Capstone Project is to carry out patent disambiguation using image data only. For this reason, the dataset that our Capstone Team has been tasked to work with includes mainly images data, and very little text data is included.

2 Business/context section

Inventorship constitutes a pivotal aspect of patent law, serving as the cornerstone for acknowledging the creative minds behind groundbreaking innovations. Identifying the rightful individuals credited with a particular invention or determining the appropriate parties to engage with for launching a business centered around a specific technology can pose significant challenges, especially when multiple individuals are attributed as inventors of said technology. The presence of different inventors for similar patents is a natural occurrence, as each contributor brings forth novel contributions to their respective technological domains. However, the complication escalates when different inventors share identical names, further exacerbating the intricacy of the situation.

Recognizing this inherent complexity, we advocate for the development of technological solutions capable of discerning between patent inventors based on the the images they use in their work. Such advancements hold the potential to revolutionize the landscape of patent law by expediting the process of inventorship determination and facilitating automated scrutiny for plagiarism across various mediums such as CAD designs, images, and research findings. These transformative breakthroughs, if proven reliable, could serve as invaluable tools in resolving legal disputes and fostering greater efficiency within the patent ecosystem.

3 Technical Introduction

In this section, we delve into the specifics of our task and detail the construction of our model. We commence by providing a mathematical formulation of the task, followed by an exposition of the machine learning and computer vision methodologies we employed or considered in our project.

3.1 Preliminary

Our task format is simple to define. For a pair of patents (P_i, P_j), given their respective image sets $I_i = \{I_1^i, I_2^i, \ldots, I_m^i\}$ and $I_j = \{I_1^j, I_2^j, \ldots, I_n^j\}$, the task requires a binary classification algorithm to map the two sets to $r \in \{0, 1\}$, where 0 represents that the two patents come from different inventors, and 1 otherwise.

The feature vector $x_{i,j}$ with respect to each data instance (P_i, P_j) can be represented as follows,

$$x_{i,j} = [\max(f(\text{Cart } (I_i, I_j)), \min(f(\text{Cart } (I_i, I_j)), \text{mean}(f(\text{Cart } (I_i, I_j)), \text{var}(f(\text{Cart } (I_i, I_j))]$$

where $x_{i,j} \in R^4$,Cart represents for Cartesian product and f represents for the computer vision extractor (encoder). As shown in Fig. 2, with the feature vector defined above, we then train a classifier h

$$h : R^4 \to \{0, 1\}$$

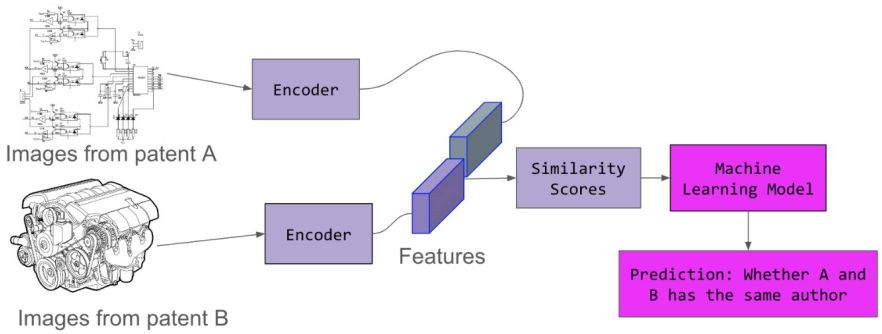

Figure 1: WorkFlow

3.2 Methodology

3.2.1 Classifier

We adopt several binary classifiers to perform patent inventor disambiguation.

- MLP (Multi-Layer Perceptron): MLP is a type of artificial neural network that finds extensive use in various machine learning tasks, including classification and regression. The name "multi-layer perceptron" stems from its structure. It comprises multiple layers of nodes (also known as artificial neurons), which are interconnected. These layers include an input layer, one or more hidden layers, and an output layer. Each node in a layer is connected to every node in the subsequent layer, forming a dense network. MLPs aim to simulate the functionality of the human brain by using interconnected perceptrons. A perceptron is the fundamental building block. It consists of three units: Sensory Unit (Input Unit), which receives input features; Associator Unit (Hidden Unit), which processes information and learns complex patterns; Response Unit (Output Unit): which produces the final output. The layers of perceptrons work together to transform input data into meaningful predictions. MLP trains its model iteratively. In each iteration, partial derivatives of the loss function are computed with respect to the model parameters. These derivatives guide the parameter updates. In addition, regularization techniques (such as L1 or L2 regularization) can be applied to prevent overfitting.

- Random Forest: Random forest classifiers are a powerful ensemble learning technique that combines multiple decision trees to improve prediction accuracy and reduce overfitting. Each decision tree in the forest is trained on a different subset of the training data and makes predictions independently. The final prediction is determined by combining the predictions of all the individual trees, typically through majority voting or averaging. Random forests are known for their robustness, ability to handle high-dimensional data, and resistance to noise. They are widely used in various machine learning applications, including classification, regression, and feature selection. The hyperparameters of random forests, such as the number of trees, the maximum depth of each tree, and the minimum number of samples required to split a node, can be tuned to optimize performance for specific datasets.

- SVM (Support Vector Machine): Support vector machines (SVMs) are a powerful supervised learning algorithm used for both classification and regression tasks. They work by finding the optimal hyperplane that separates the data points into their respective classes. The hyperplane is chosen to maximize the margin, which is the distance between the hyperplane and the closest data points from each class. SVMs are known for their ability to handle high-dimensional data, their robustness to noise, and their efficiency in training. They are also capable of performing non-linear classification by using the kernel trick, which maps the data into a higher-dimensional feature space where a linear separation is possible. The hyperparameters of SVMs, such as the kernel function, the regularization parameter, and the tolerance for constraint violations, can be

tuned to optimize performance for specific datasets. SVMs have been successfully applied to a wide range of machine learning problems, including text classification, image recognition, and bioinformatics.

3.2.2 Image Feature Extractor

To obtain information from patent images, we are going to use a computer vision feature extractor. Computer vision feature extractor is a technology used to extract useful information from images. It can be used for image recognition, classification, segmentation and other tasks. Feature extractors can be divided into two major categories: classic extractors and deep learning feature extractors.

Classic feature extractors are based on hand-designed algorithms, and they usually utilize some basic attributes of images, such as color, texture, shape, edges, etc., to construct feature vectors or feature descriptors. The advantages of these feature extractors are simple calculation, fast speed, and do not require a large amount of training data. However, the disadvantages are that the feature expression ability is limited, not robust enough, and difficult to adapt to complex and changeable scenarios. The classic, non-deep-learning feature extractors we use are:

- SSIM (Structural Similarity Index Measure)[3]: SSIM is an index used to measure the structural similarity between two images. It considers the brightness, contrast and structure of the image and can be used for image quality assessment, image enhancement, etc. application.
- SIFT (Scale Invariant Feature Transform)[4]: SIFT is an algorithm used to detect and describe key-points in images. It is robust to scale and rotation and can be used for image matching, object recognition, three-dimensional reconstruction, etc.
- LBP (Local Binary Pattern)[5]: LBP is an algorithm used to describe texture features in an image. It divides the image into small areas, then combines the pixels in each area with the pixels in its neighborhood to obtain a binary pattern. LBP is widely used in face recognition, expression analysis, and texture classification.
- HOG (Histogram of Oriented Gradient)[6]: HOG is an computer vision algorithm used to describe shape features in an image. It divides the image into small areas, and then calculates the gradient direction and amplitude in each area to obtain a directional histogram. It can be used for human detection, pedestrian tracking, vehicle identification and other applications.

Deep learning feature extractors are neural network-based models that can automatically learn feature representations from a large amount of annotated data without the need to manually design features. The advantages of these feature extractors are strong expression ability and adaptability to

complex and changeable scenarios. However, the disadvantages are intensive computations and the need of a large amount of training data. The deep learning encoders we have used are:

- ResNet (Residual Network)[7]: ResNet is a deep convolutional neural network used for image classification. It introduces the concept of residual connection, which can effectively solve the gradient vanishing and degradation of deep networks problem. ResNet can achieve a depth of more than 100 layers, and can be used for applications such as image classification, object detection, and semantic segmentation.

- U-Net[8]: U-Net is a deep convolutional neural network for image segmentation. It has a U-shaped structure, consisting of a down-sampling encoder and an up-sampling It is composed of a decoder and uses skip connections to fuse the characteristics of the encoder and decoder. It can be used for medical image segmentation, satellite image segmentation, automatic driving and other applications.

- ViT (Vision Transformer)[9]: ViT is a deep self-attention neural network used for image classification. It divides the image into multiple small areas (patches), and then uses the pixel value of each area as a vector. Input into a transformer model, which can be used for image classification, image generation, image understanding and other applications.

Although these deep learning methods are supposed to be very helpful for the disambiguation task, the limitation of the available computational resources prevents us from applying them. To be specific, the dataset have a number of around 100 000 images but the team is not supported with GPU or any sufficient computational resources. Thus, in the data processing we mainly use the non-deep-learning feature extractors.

4 Experiment

4.1 Datasets

Our project is structured into two distinct phases. Initially, we focus on constructing a dataset, followed by training a machine learning model using this dataset. In this section, we will elaborate on the details of the dataset construction process.

To design our training and testing dataset, we firstly build a raw dataset as our baseline pool. Our baseline dataset was obtained through the merger and cleansing of various patent features datasets, and the following sections will detail the composition and sources of the baseline dataset. Initially, to collect the basic information of all the patent pairs we need, we downloaded all relevant data on patent pairs from USPTO, United States Patent and Trademark Office, and after cleansing, each

record of data represents a pair of patents, containing the unique patent numbers for both patents, their respective patent inventors, and the publication dates. After that, we added the label for each record; if the pair of patent share the same inventor we labeled it as 1, otherwise labeled as 0. On this basis, to guarantee comprehensive coverage and representativeness, we randomly selected ten points in time for model training, choosing all patent pairs published at these ten points. Then, we calculated the similarity between all images in each pair of patents using classical Python computer vision packages. Specifically, for m images in patent-a and n images in patent-b, we calculated m*n similarity scores. We utilized five algorithms to calculate the scores: LBP, HOG, SIFT, SSIM, and BF (for detailed principles, please refer to the technical section). Through these five algorithms, we derived the following similarity metrics: top1 (the highest similarity score among the two sets of patent images), top2 (the second highest similarity score among the two sets of patent images), top3 (the third highest similarity score among the two sets of patent images), avg (the average score of all image pairs within the two sets of patent images), std (the standard deviation of the scores of all image pairs within the two sets of patent images), and worse (the lowest score of all image pairs within the two sets of patent images). Therefore, we obtained a total of 30 features to describe the similarity between the two sets of patent images using five algorithms.

Additionally, to construct a base dataset, we downloaded the disciplinary information for each data entry from the USPTO and added a column variable to the dataset to determine whether a pair of patents belongs to the same discipline. If is_same_cpc equals 0, it indicates that the two patents do not belong to the same discipline. Conversely, if is_same_cpc equals 1, it signifies that the two patents are within the same discipline. By incorporating this column of data, we aim to assess whether images can aid in determining whether two patents belong to the same discipline.

Therefore, we designed a dataset for training. To easily differentiate the instances with the same inventors or not, we included 1000 records with a half positive instances(have the same inventors),and another half negative instances(have different inventors).

Finally, we constructed a test set. In order to get a balanced dataset and easily separate four different occasions :with the same inventors and in the same domain, with the same inventors but not in the same domain, have different inventors but in the same domains, and have different inventors and in different domains. We designed a dataset with four quarters, with 1000 records in total.

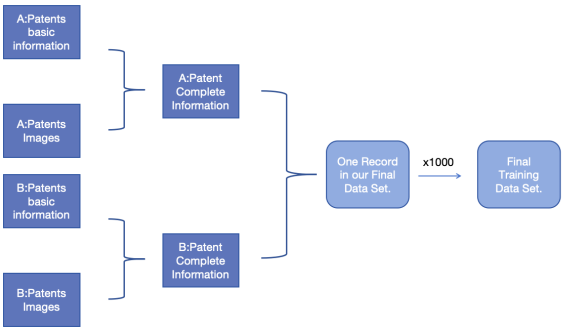

Figure 2: Data Set Build Process

4.2 Metrics

4.2.1 Accuracy

In machine learning, the accuracy score is a common metric used to evaluate the performance of a classification model. The accuracy score is calculated as:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Samples}}$$

The accuracy score represents the proportion of correct predictions made by the model. A higher accuracy score indicates that the model is more successful in correctly predicting the class labels for the given dataset. It is often expressed as a percentage, ranging from 0% (no correct predictions) to 100% (perfect accuracy). Remember that accuracy alone may not be sufficient for all scenarios, especially when dealing with imbalanced datasets or when different classes have varying importance. In such cases, other metrics like precision, recall (TPR), F1-score, or balanced accuracy may provide a more comprehensive view of model performance.

4.2.2 TPR and FPR

TPR, also known as recall, measures the proportion of positive instances (actual positive samples) that the model correctly identifies. It answers the question: "Out of all actual positive cases, how many did the model predict correctly?" The formula for TPR is:

$$\text{TPR} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

FPR, on the other hand, quantifies the proportion of negative instances (actual negative samples) that the model incorrectly classifies as positive. It answers the question: "Out of all actual negative cases,

how many did the model mistakenly predict as positive?" The formula for FPR is:

$$\text{FPR} = \frac{\text{False Positives (FP)}}{\text{False Positives (FP)} + \text{True Negatives (TN)}}$$

TPR (recall) focuses on the model's ability to correctly identify positive cases, which is crucial in scenarios where missing positive instances is costly (e.g., disease diagnosis). FPR highlights the model's tendency to produce false alarms (incorrectly labeling negatives as positives). Minimizing FPR is essential when false positives have significant consequences (e.g., spam detection).

4.2.3 F1 Score

The F1 score is a machine learning evaluation metric that combines precision and recall scores. It assesses the predictive skill of a model by elaborating on its class-wise performance rather than an overall performance (as done by accuracy). Unlike accuracy, which computes how many times a model made correct predictions across the entire dataset, the F1 score focuses on class-wise performance. The F1 score is the harmonic mean of precision and recall. It is calculated as follows:

$$F1 \text{ Score} = \frac{2 \cdot (\text{precision} \cdot \text{recall})}{\text{precision} + \text{recall}}$$

Precision evaluates the model's accuracy in making favorable predictions (correctly identifying positive cases). It measures how many of the "positive" predictions made by the model were correct. Recall measures how many of the positive class samples present in the dataset were correctly identified by the model. The F1 score balances precision and recall, offering a trade-off between the two. It is particularly useful when dealing with imbalanced datasets or scenarios where missing positive instances (false negatives) or false alarms (false positives) have significant consequences.

4.2.4 AUROC

The Area Under the Receiver Operating Characteristic (AUROC) curve, also known as the ROC curve, is a graphical representation of the performance of a binary classification model at various classification thresholds. It assesses the ability of the model to distinguish between two classes: typically the positive class (e.g., inventorship, presence of certain property) and the negative class (e.g., absence of certain property). The AUROC measures the probability that the model will assign a randomly chosen positive instance a higher predicted probability compared to a randomly chosen negative instance. It quantifies the model's ability to distinguish between the two classes. When training a ML model, the goal is to maximize this area to achieve the highest TPR (sensitivity) and lowest FPR (specificity) at the given threshold. AUROC is particularly useful when: 1) evaluating the capability of a discriminative model.

2) dealing with imbalanced datasets or scenarios where false positives and false negatives have varying consequences.

4.3 Results and Interpretations

4.3.1 Model evaluation while as a whole

Model	ACC ↑	TPR ↑	FPR ↓	F1 ↑	AUROC ↑
SVM	67%	60%	26%	65%	67%
RF	74%	69%	22%	72%	74%
MLP	66%	58%	25%	63%	66%

Table 1: Model performance on the whole dataset

Tab. 1 presents the performance of our models when tested on the whole dataset. Random Forest (RF) has the best performance in terms of all metrics. We believe this is due to its stronger ability of capturing data pattern than SVM, and more robustness than MLP.

4.3.2 Model evaluation while factoring in field

Model	ACC ↑	TPR ↑	FPR ↓	F1 ↑	AUROC ↑
SVM	61%	48%	26%	55%	61%
RF	69%	61%	23%	67%	69%
MLP	63%	51%	24%	58%	63%

Table 2: Model performance within the same field

Model	ACC ↑	TPR ↑	FPR ↓	F1 ↑	AUROC ↑
SVM	73%	72%	26%	73%	73%
RF	78%	77%	21%	78%	78%
MLP	70%	65%	26%	68%	70%

Table 3: Model performance within different fields

Tab. 2 and 3 present how the model performs when tested in subsets of the dataset where inventors belong to the same field or not, respectively. Again the Random Forest model shows better performance, just as in Tab. 1. Additionally, it should be noted that the model performs much better when asked to differentiate when the patents belong to different fields. We attribute these higher performance to the hypothesis that patents from the same domain will be more difficult to tell apart since they tend to use similar images (e.g. same kind of graph), whereas with patents from different fields, images will not look similar naturally.

5 Discussion

In this paper, we focused on exploring whether it is feasible to predict if two patents are written by the same inventor or not utilizing only the images from said patents. We used non deep learning feature extractor algorithms coupled with various ML classifiers in order to train a model on a dataset of patent pairs. Our first results showed better than random performance, at 74% accuracy for our random forest model, which implied that the images hold information relevant to the task, but that either said information or our methods were insufficient to produce a highly accurate model. Following these results we decided to test the hypothesis that part of the similarity between two images from patents written by the same inventor can be attributed to both patents belonging to the same field of study. While testing this hypothesis, we've observed that our model performed better when working on patents from different fields rather than when both patents belonged to the same field. This result suggests that distinguishing inventors within the same domain is more challenging, and this is possibly due to the fact that patents from a certain domain will use visual aids related to said domain, like how chemistry patents are very likely to depict molecular representations.

Having said that, our research has shown the potential of using image data for patent inventor disambiguation. We set out to provide a perspective on patent differentiation that is distinct from text based approaches, and are confident that with more work, this new perspective could prove useful in patent law and related fields. As for future work, we believe it could be interesting and might even bring good results to try calculating the similarity scores with deep learning based feature extraction, this would require however significant resources in the form of graphical computing power.

References

[1] T. VI, W. M, S. DR, and S. NR, "A probabilistic similarity metric for medline records: a model for author name disambiguation," *AMIA Annu Symp Proc*, 2003.

[2] N. R. S. Torvik Vetle I, "Author name disambiguation in medline," *Association for Computing Machinery*, 2009.

[3] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[4] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[5] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[6] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1. Ieee, 2005, pp. 886–893.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[8] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 2015, pp. 234–241.

[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.