**CS445 NLP Final Report**

**Kanat Özgen, Erdem Özcan, Mert Coşkuner, Gürsel Yiğit Pekgöz**

## **1. Introduction**

Intent detection is an important field in NLP where systems are created in order to predict the intention behind a given string. This can be useful in mobile applications, customer support, and security applications.

Our solution revolves around incrementing the accuracy of a given system, which in the case of the NAACL18 paper named Slot-Gated Modeling for Joint Slot Filling and Intent Prediction. In this paper, Goo et al. procured a methodology and an architecture that implements Bidirectional LSTMs and neural networks to predict the slot fillings and intents in the ATIS dataset. Since the paper was old, we had to fix it to make it meet the dependency requirements. After doing so, we found openings that could lead to making the model more accurate. We used a variety of methods to increase the accuracy of the model. We also incorporated a procedure for calculating the f1 measures in the original code, which they did not provide. We were able to increase the accuracy of the system by more than one percent and increase the f1-measure by more than ten.

## **1.1 Dataset**

We used the ATIS dataset, which stands for Airline Travel Information System. The dataset consists of BIO taggings of entities within the dataset as well as labels that indicate the intent of a user. The original split contains 4478, 500, and 893 intent-labeled reference utterances in train, development, and test set respectively. There was an abundance of resources that revolved around this dataset, and we considered that these could be good starting points in creating an even more robust system.

**2. Trials Before the Chosen Paper**

The precision-recall curves, classification reports, and confusion matrix are in the appendices section.

### **2.1 Baseline Model (Naive Bayes)**

This model indicated imbalanced performance across classes. Frequent classes like "flight" show high accuracy and F1-score, while rare classes like "city" and "restriction" perform very poorly. The low macro average (precision: 0.41, recall: 0.36) shows the imbalance.

### **2.2 Logistic regression with TF-IDF / Bert**

In this section, the baseline model was updated and the data was trained with logistic regression with TF-IDF and Bert separately the results were as follows.

Logistic regression with TF-IDF showed no significant improvement and indicated imbalanced performance across classes. Frequent classes like "flight" show high accuracy and F1-score, while rare classes like "city" and "restriction" perform very poorly. The low macro average (precision: 0.59, recall: 0.52) highlights this imbalance. Therefore, the model was updated with Bert instead of TF-IDF. Furthermore*,* Logistic regression with Bert also showed no significant improvement to our baseline model and logistic regression with TF-IDF. Also, again the model showed the imbalanced performance across classes. Therefore, the model was updated in the following sections. There was a significant problem with missing data in the classes.

### **2.3 N-gram Model Comparisons**

We experimented with three models: unigram, bigram, and trigram, using a bag-of-words approach with TF-IDF vectorization. Among these, the Bigram model performed the best, achieving 93.73% accuracy, a weighted F1-score of 92.39%, and a macro average F1-score of 57.00%. The unigram model followed closely, while the trigram model's performance dropped significantly due to data sparsity, with a macroaverage F1-score of 48.00%. These results highlight that the Bigram model effectively balances context capture and computational efficiency, making it superior for intent classification.

### **2.4 Oversampling and Overall Model Performance**

We applied oversampling to balance all classes in the dataset to 3666 examples, aiming to improve fairness across intents. While this strategy improved the representation of minority intents such as airfare+flight and meal, it did not lead to an overall performance boost. In fact, the accuracy slightly dropped from 93.73% to 93.06%, and the weighted F1-score also saw a minor decrease. Despite these general declines, certain underrepresented intents showed noticeable improvements in both metrics. Our analysis confirmed that while oversampling enhanced performance for a few classes, it introduced minor degradation in others.

**2.5 Neural Network with Bert / TF-IDF**

We trained the data with the neural network using Bert and TF-IDF separately. NN with using TF-IDF showed no improvements and performs poor in rare classes like “distance”, “meal”, “ground\_fare” it slightly improves frequent classes like “flight” but nothing significant compared with baseline model. NN with using Bert has show significant improvement for almost all classes. It has a high accuracy of %98 and high f1-scores 0.72 and 0.98(macro average, weighted average). These result shows that neural network using bert effectively balances classes with both frequent and rare classes effectively for intent detection.

## **3. Methodology for Chosen Paper**

### **3.1 Chosen Paper**

For this task, the NAACL18 paper named Slot-Gated Modeling for Joint Slot Filling and Intent Prediction was chosen. The reason why this paper was selected can be broken down into three parts:

Their research mainly focused on slot filling. Therefore, we decided to improve the system such that it mainly focuses on intent. We reduced the effectiveness of slot-filling loss in the multi-way loss function by introducing the following formula:

Where the alpha variable is a value between 0 and 1 that dictates the importance of slot loss within the loss function. We established this function and set the alpha to a very low value, therefore eliminating the importance of slot loss within the whole loss function. Due to the fact that slot-to-intent gating was important in the way of gathering signals that might indicate intents, we wanted it to be a value different than 0.

The dataset is extremely sparse. The atis\_flight label dominates the label space by a huge margin. We detected that this paper falls short of smoothing out the label space, which turned out to be a good spot for increasing the accuracy of the system. We wrote a procedure such that for each gold label, we put (1 - smoothing) at the correct index and distribute smoothing / (num\_classes - 1) among the other indices. This helps reduce overconfidence and might improve generalization.

They used softmax instead of a proper sequence prediction algorithm such as HMM or CRF. We decided to use CRF to improve the architecture and therefore the accuracy. One might consider why we decided to incorporate an improvement regarding the slot-filling problem. This was because slot fillings provided signals to the system via the slot-to-intent gating layer which provides important information about the intents.

### **3.2 Training Details**

The system was trained with the original 4478, 500, and 893 splits taken into consideration; which in turn amounts to a train/dev/test split of approximately 70/10/15. Minimal preprocessing was done, aside from extracting the dataset pickles into IOB and RASA JSON formats for processing. We have devised an extra hyperparameter for the system which is the aforementioned alpha value that determines the rate at which the slot loss is incorporated into the calculation of the overall loss. This turns out to be a hyperparameter because the system uses slot-to-intent gating to determine the intent probabilities, thus making its complete annihilation redundant and unnecessary. A sweet spot of the alpha value is to be determined, considering the novel improvements over the architecture regarding the loss function.

Epoch size was determined to be 280, which is the value that we kept using. The model did not include many hyperparameters, which is why there was no extensive hyperparameter tuning. A specific epoch count of 20 was established, but the patience value as a hyperparameter holds importance since we have changed the behavior of early stopping to only consider the loss of intent.

### **3.3 Implementation**

For the task at hand, due to the necessity to initially replicate the paper, we needed to use the Python version of 3.5.6 as well as the tensorflow version of 1.4. This turned out to be a difficult task since the tensorflow 1.4 was no longer supported. The “\_linear” attributes were removed from the later versions of tensorflow and thus, we needed to fix the code to make it worthwhile. We used the “Dense” class from Tensorflow to replace the layers with the dense layers. Since we incorporated this both in our improved approach and theirs, negligible effects on the methodology are expected. The model was trained on an Nvidia GeForce GTX 1080 GPU with 8 gigabytes of VRAM, to increase the speed of attention-seeking and tensor multiplications.

As it was told in previous chapters, we included the implementation of Conditional Random Fields, which is an algorithm used with the same reason as the Hidden Markov Models, to predict slots, instead of the softmax function that they had implemented. As told previously, slot filling holds importance in determining the intents.

## **4. Results**

Our improved implementation surpassed the accuracy and the macro F1-score of both the Naive Bayes baseline and the NAACL18 paper. Due to the nature of the codebase, we are unable to provide the PR curves for our implementation but we have the confusion matrix and the classification report available within the appendix of this report.

## **5. Discussion**

### **5.1 Dataset Impact**

The dataset is very widely referenced, therefore it leads us to create a robust architecture. This can be corroborated by the fact that most of the papers in the papers with code database regarding the intent detection field are reliant on this dataset and this dataset serves as a baseline for other intent detection tasks, such as customer support. While SNIPS is also a good dataset to begin with, some encoding issues come with that dataset and it is not as widely used and recognized as ATIS. ATIS also possesses important metadata such as BIO tagging which helps extensively.

### **5.2 Methodological Analysis**

Our method falls short in evaluating the slot fillings, which is also an important task. Therefore, it can be said that our method is not universal. However, it ranks as the fourteenth in the paperswithcode database, surpassing even the accuracy of the LIDSNet architecture.

### **5.3 Potential Improvements**

Potential improvements over the system could be hyperparameter tuning over the alpha variable, which can increase the accuracy of the system even more. Also, in the scope of this model, the embeddings are calculated on the go. Embedding models such as BERT could be used to get more accurate embeddings, thus increasing the model performance. We have discussed whether to use BERT embeddings in our method, but we discarded it due to the fact that domain-agnostic embeddings may be detrimental to the overall system performance given the constrained vocabulary size and small context of utterances.

**6. Conclusion**

In this report, we successfully enhanced the intent detection system initially proposed by Goo et al. in their NAACL18 paper on Slot-Gated Modeling for Joint Slot Filling and Intent Prediction. By addressing outdated dependencies and implementing strategic modifications—such as adjusting the loss function to prioritize intent prediction, incorporating label smoothing to balance the label space, and replacing the softmax layer with Conditional Random Fields—we achieved significant improvements in both accuracy and macro F1-score on the widely recognized ATIS dataset. Our refined approach not only surpassed the original model and a Naive Bayes baseline but also demonstrated competitive performance within the broader intent detection landscape, ranking fourteenth in the paperswithcode database.

While our focus was primarily on enhancing intent prediction, leaving slot-filling evaluation as an area for future exploration, the methodological advancements presented lay a robust foundation for further development. Potential avenues for improvement include hyperparameter tuning of the alpha variable and integrating advanced embedding techniques like BERT, which could further elevate model performance. Overall, our contributions underscore the value of targeted architectural refinements and highlight pathways for building more accurate and resilient intent detection systems applicable to diverse real-world applications such as mobile applications, customer support, and security.

## **7. References**

Coucke, Alice, et al. "Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces." *arXiv preprint arXiv:1805.10190* (2018).

Goo, Chih-Wen, et al. "Slot-gated modeling for joint slot filling and intent prediction." *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. 2018.

Agarwal, Vibhav, et al. "Lidsnet: A lightweight on-device intent detection model using deep siamese network." *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2021.

## 

## 

## 

## 

## 

## 

## 

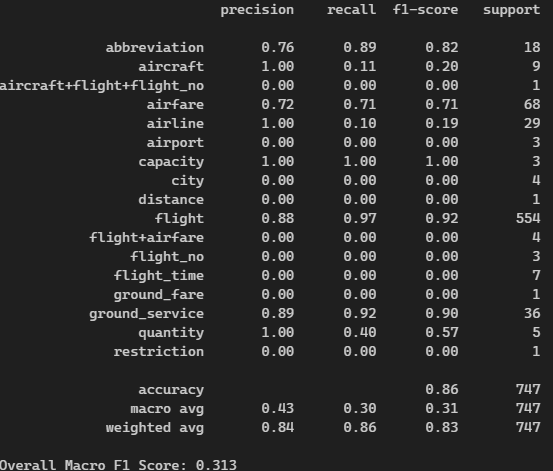
## 

## 

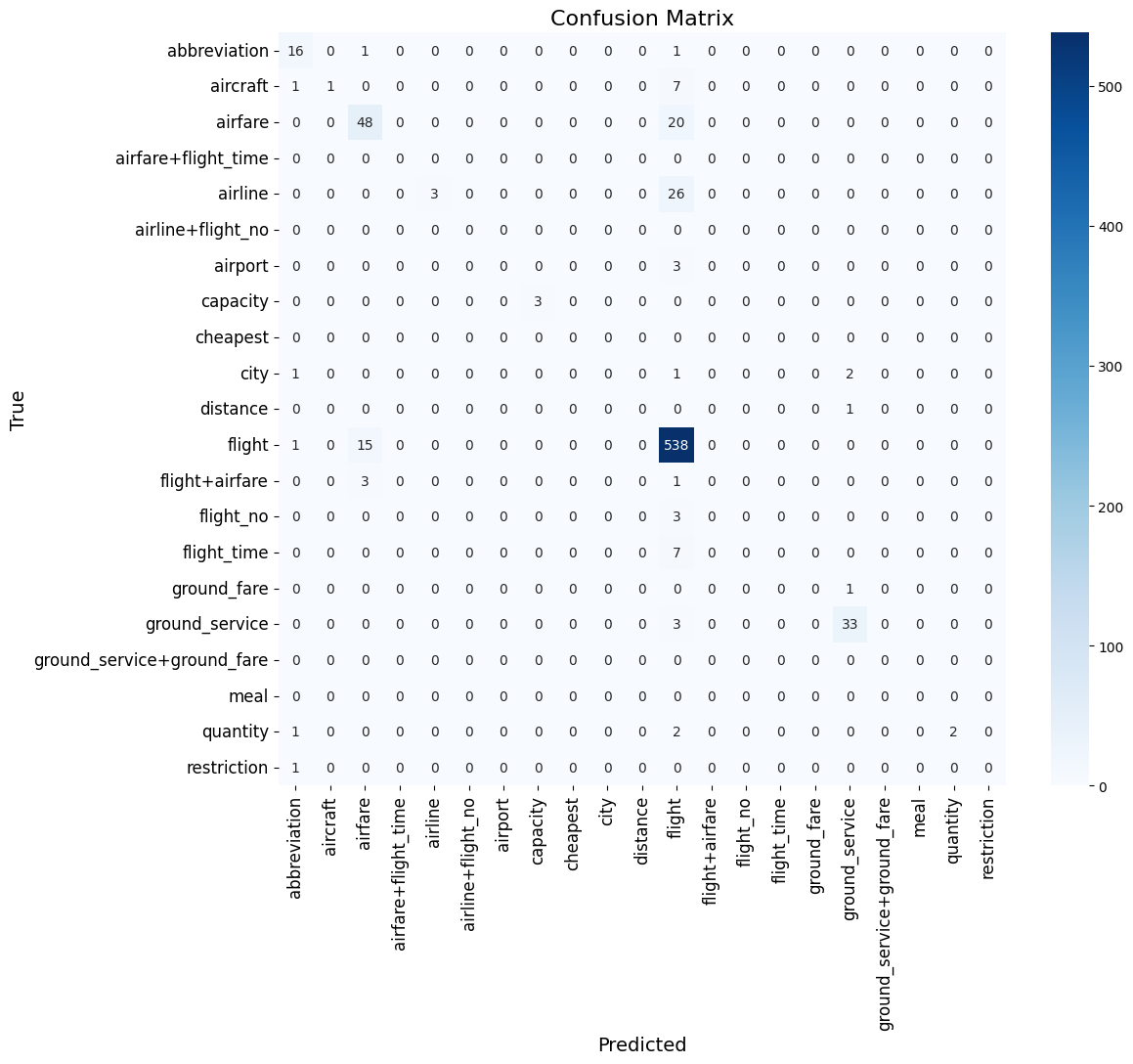
## 

## 

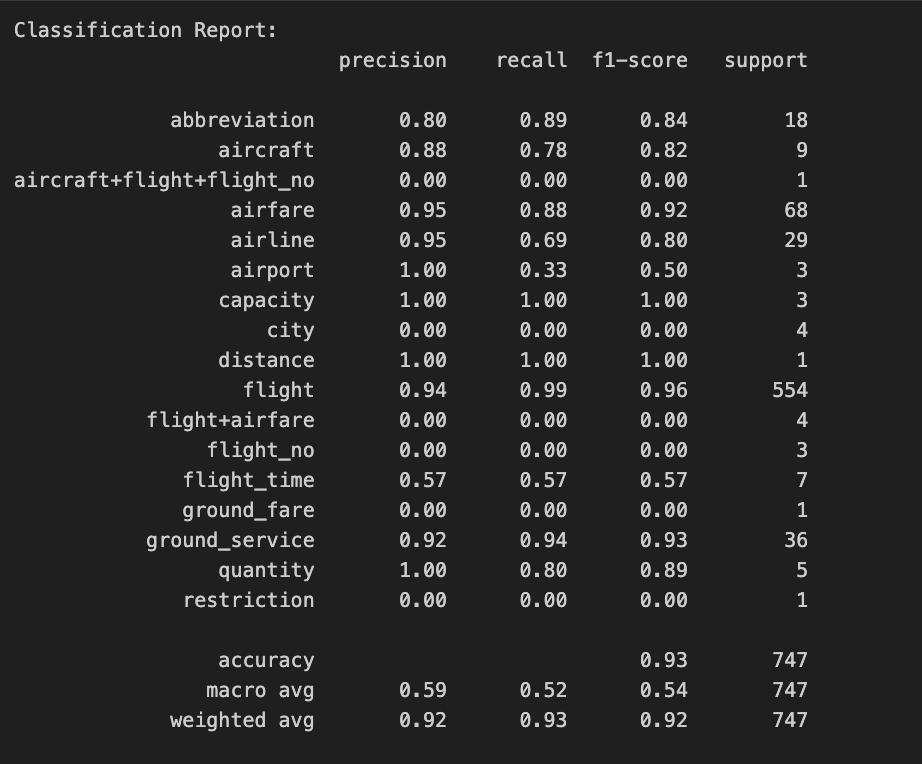
## **8. Appendix**



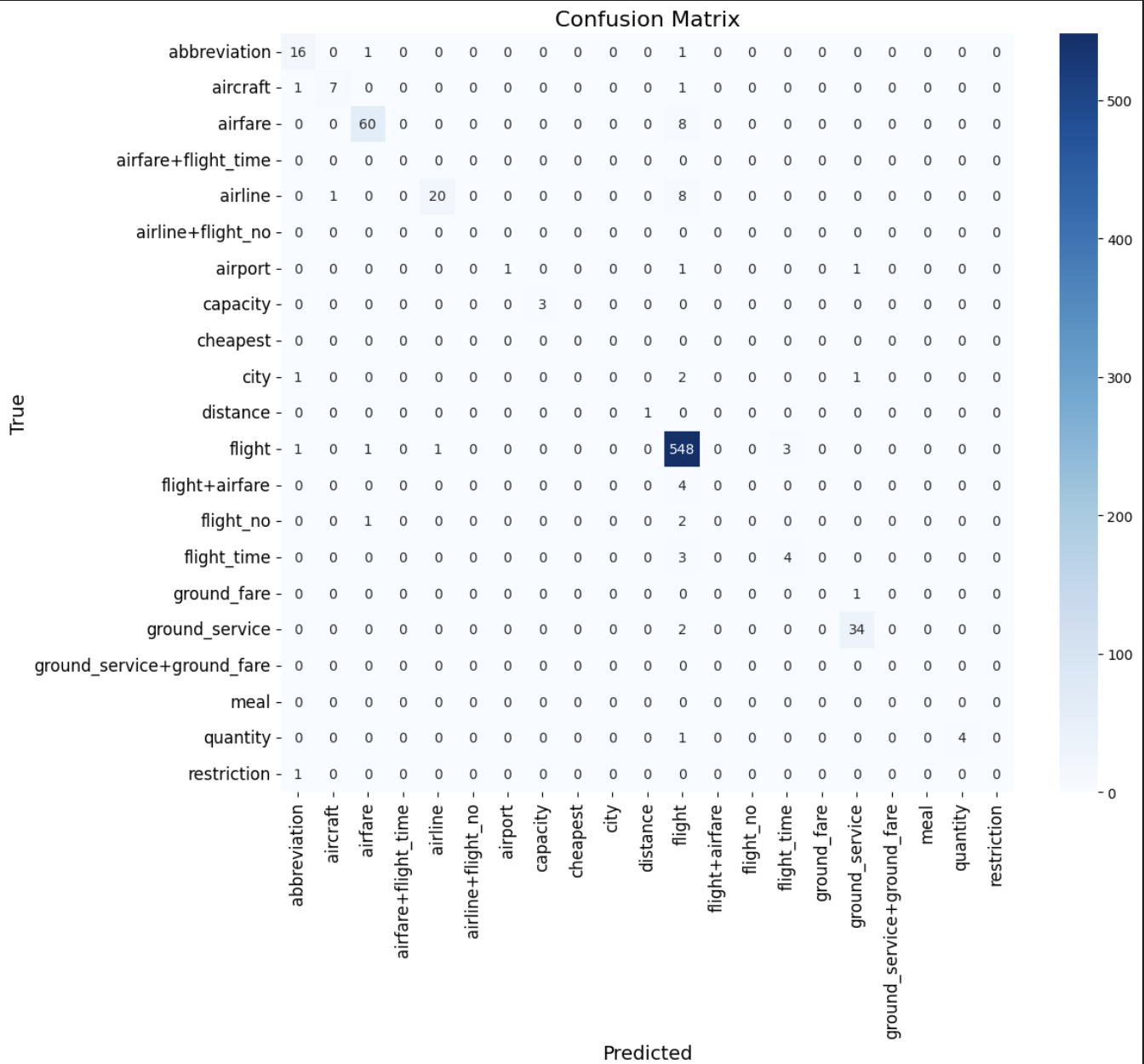
*Figure 1: Baseline Model Classification Results*



*Figure 2: Baseline Model Confusion Matrix*

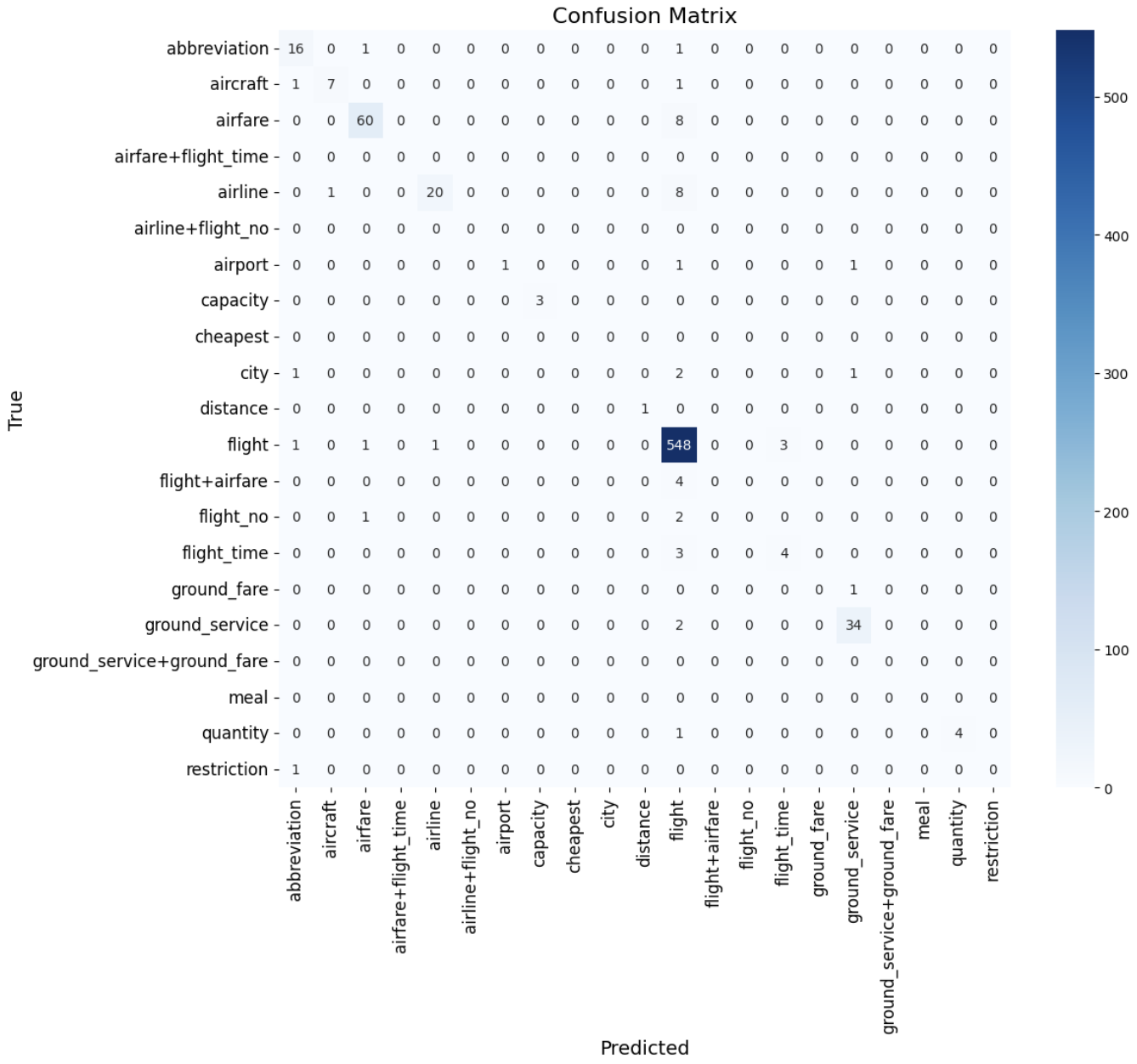
****

*Figure 3: Logistic regression with TF-IDF Classification Report*

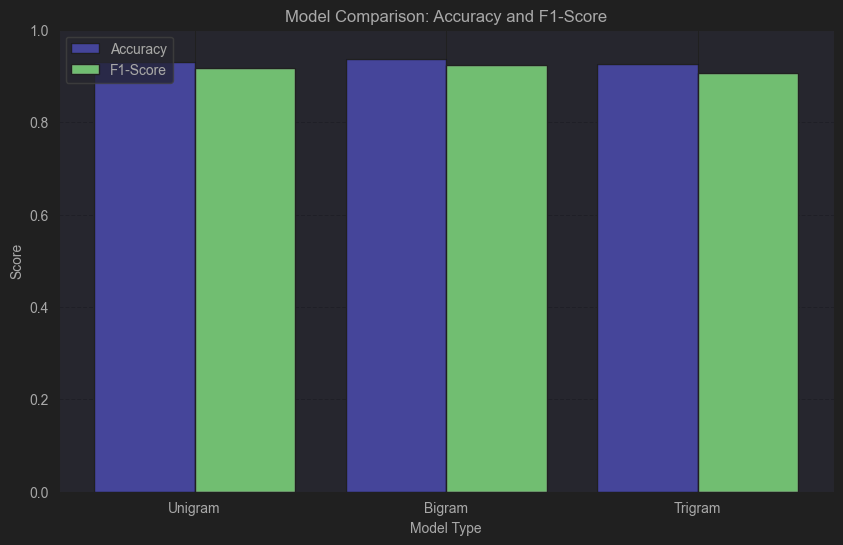
*****Figure 4: Logistic regression with TF-IDF Confusion Matrix*

### 

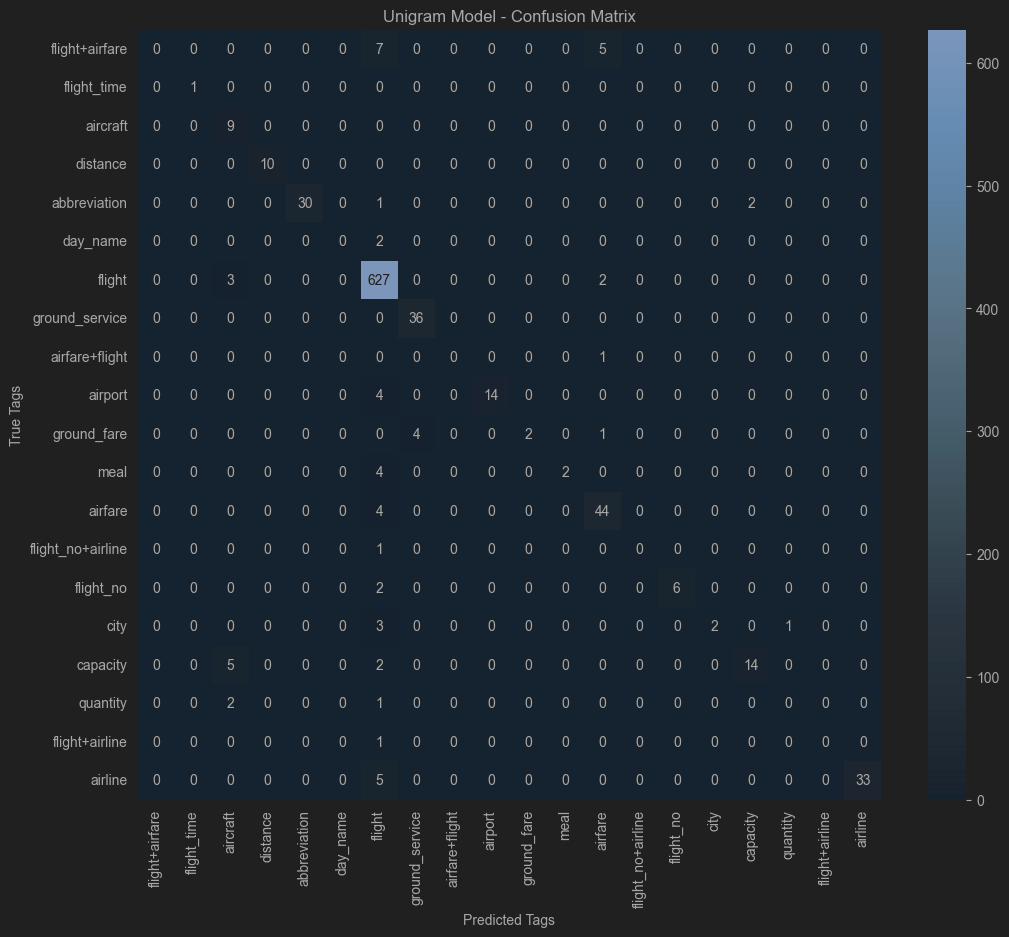
*Figure 5: Logistic regression with Bert Classification Report*

**

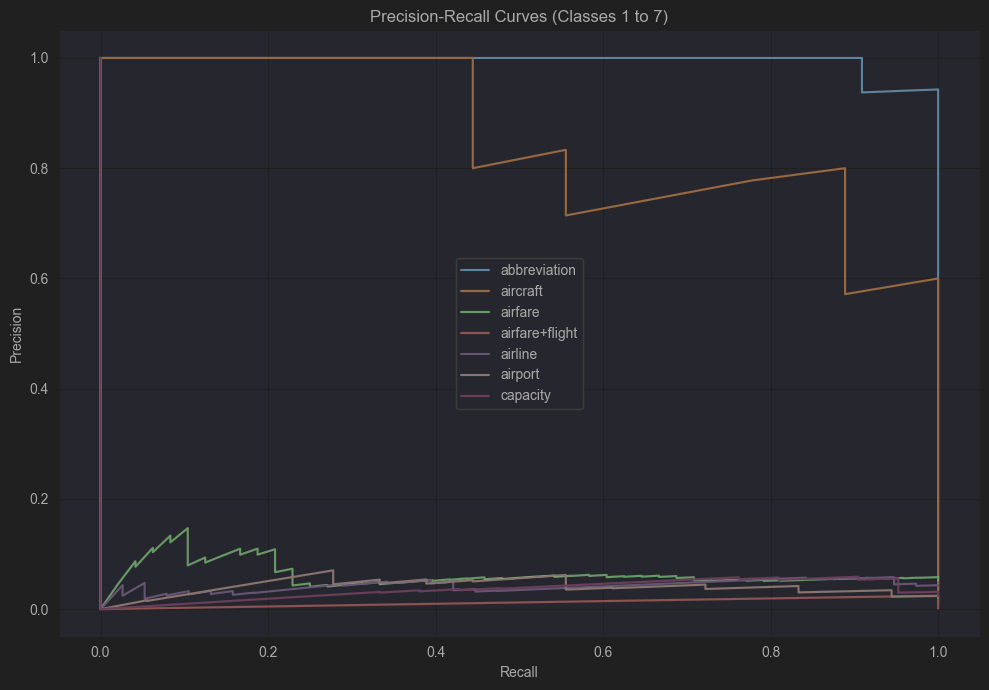
*Figure 6: Logistic regression with Bert Confusion Matrix*



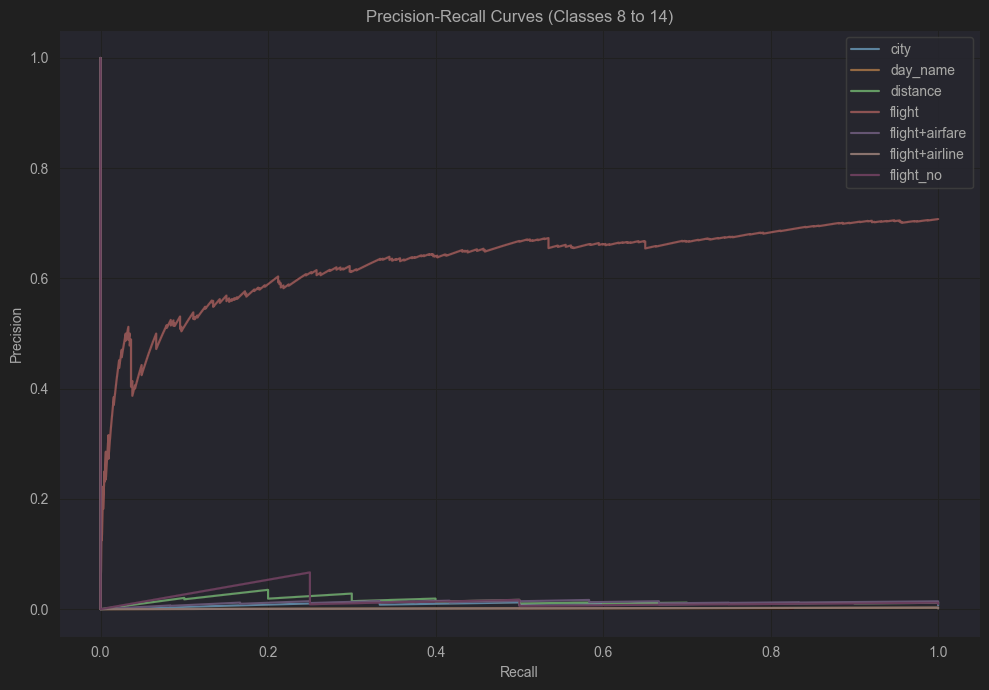
*Figure 7: Comparison of N-Gram Models.*



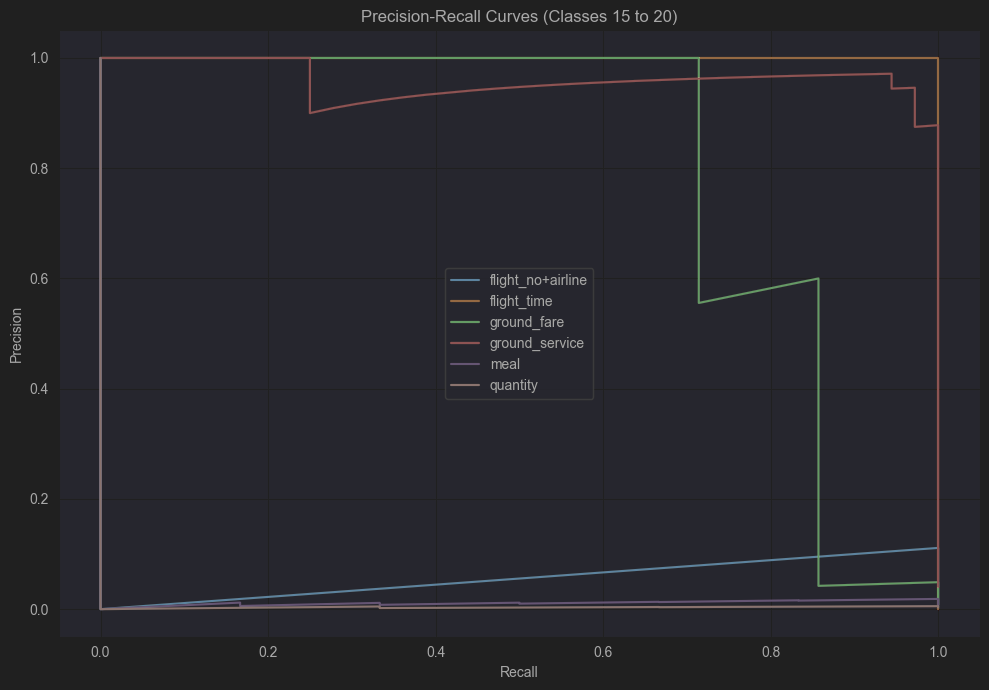
*Figure 8: Confusion Matrix of the Unigram Model*

**

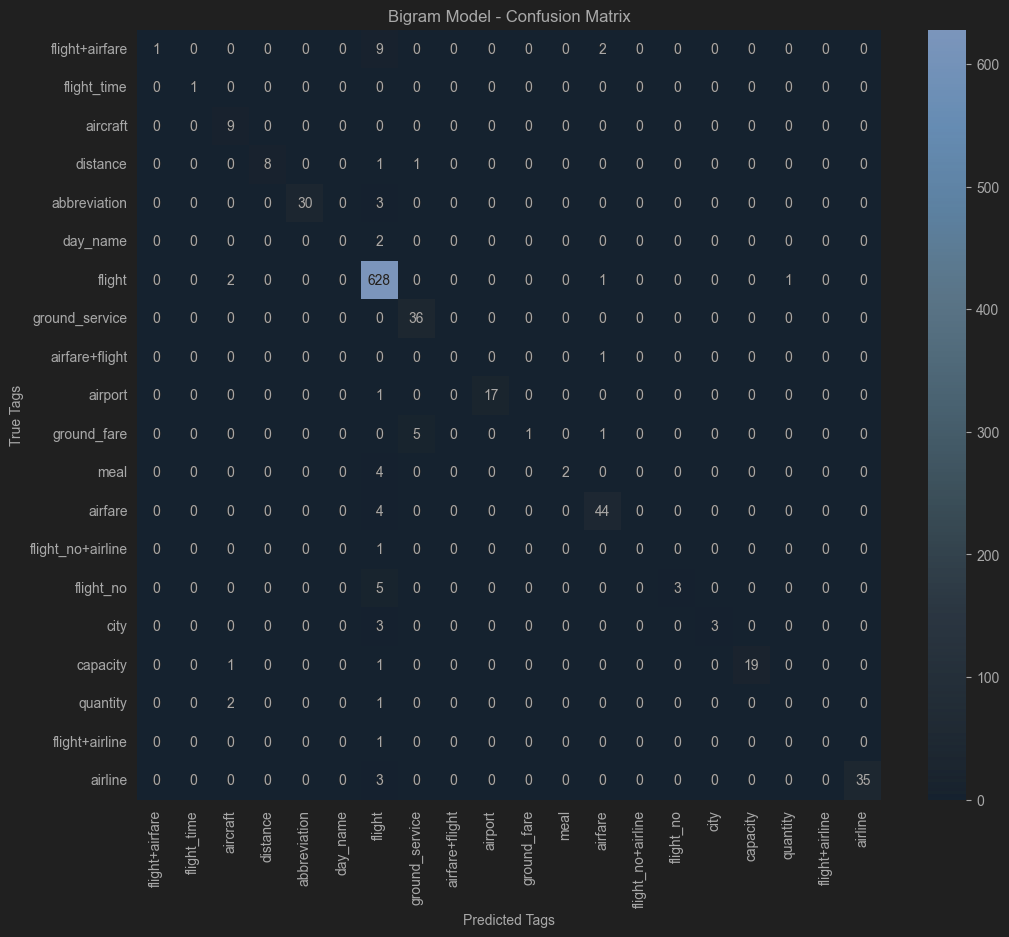
*Figure 9:* Precision-Recall curves *of the Unigram Model*

**

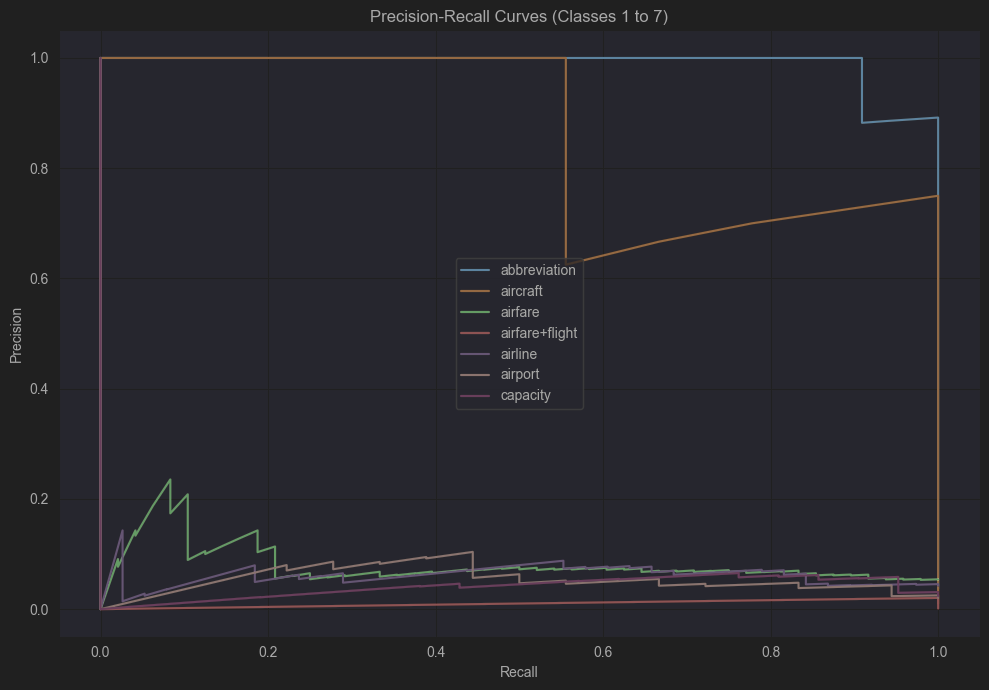
*Figure 10:* Precision-Recall curves *of the Unigram Model*

**

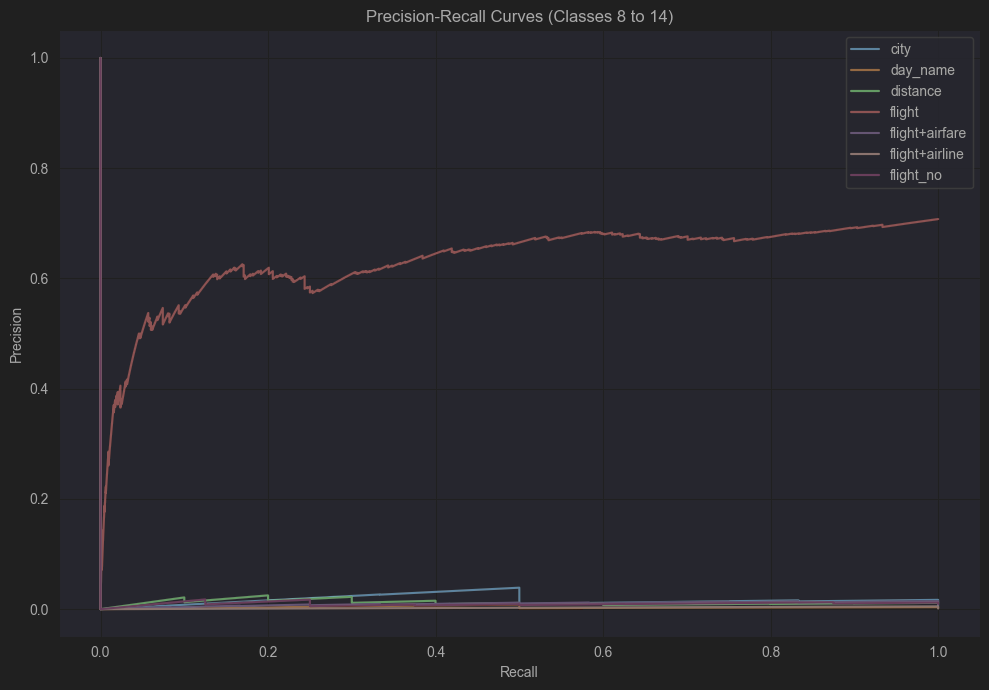
*Figure 11:* Precision-Recall curves *of the Unigram Model*



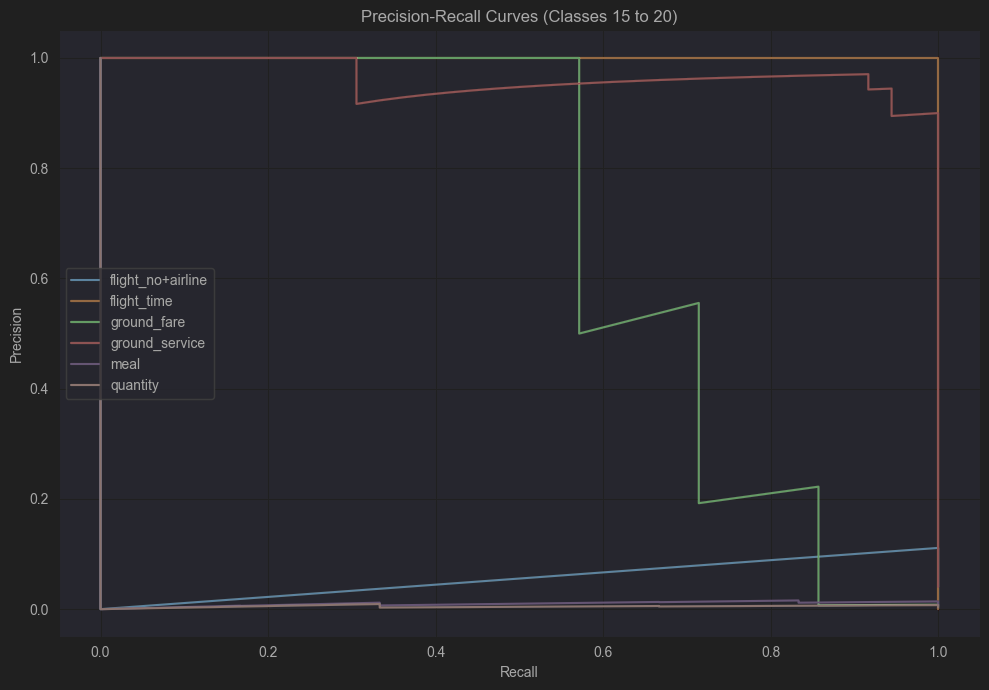
*Figure 12: Confusion Matrix of the Bigram Model*



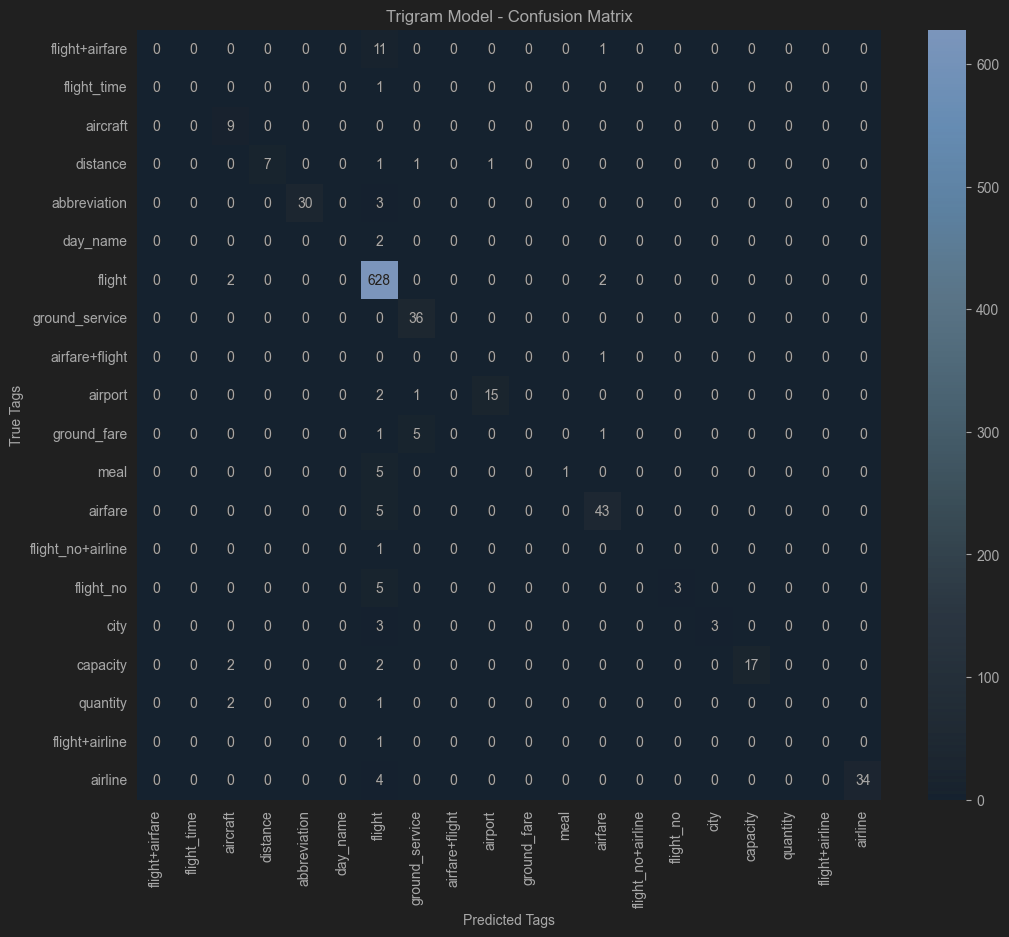
*Figure 13:* Precision-Recall curves *of the Bigram Model*

**

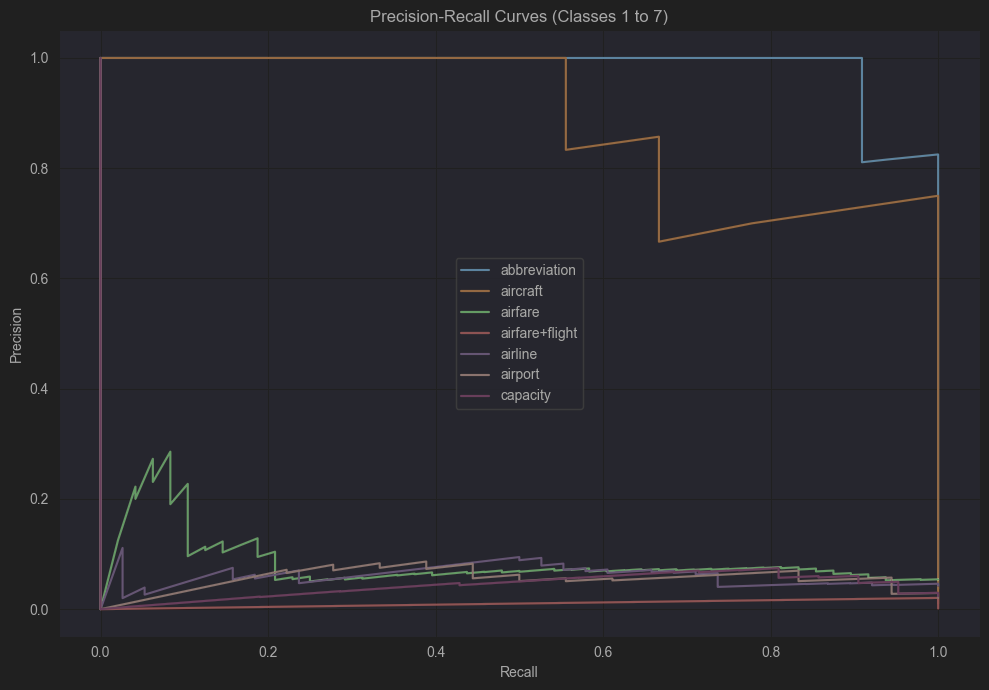
*Figure 14:* Precision-Recall curves of *the Bigram Model*

**

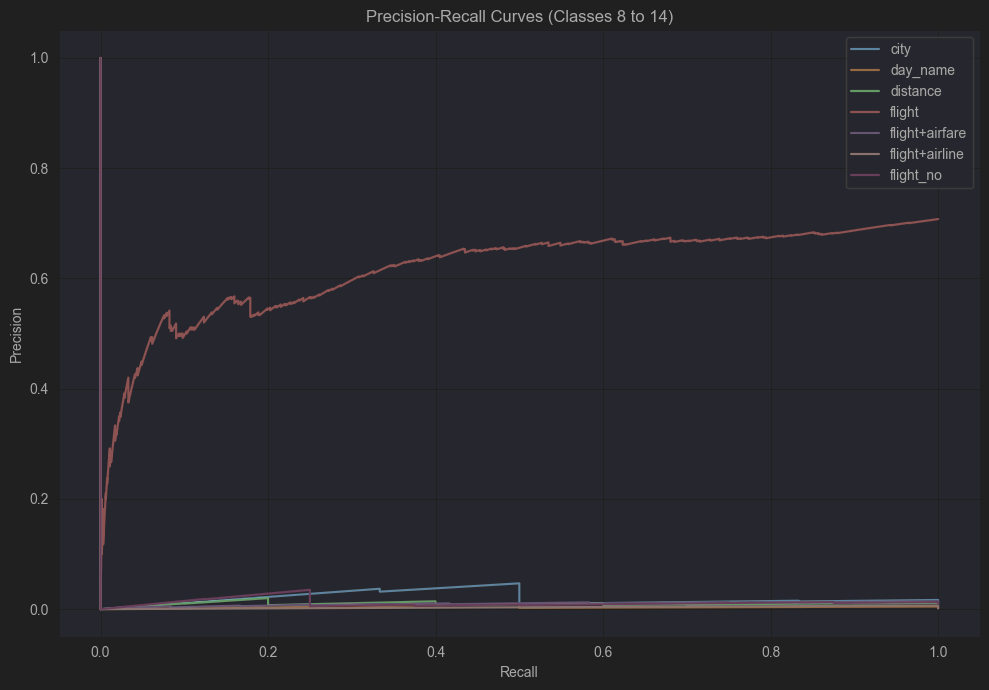
*Figure 15:* Precision-Recall curves of *the Bigram Model*



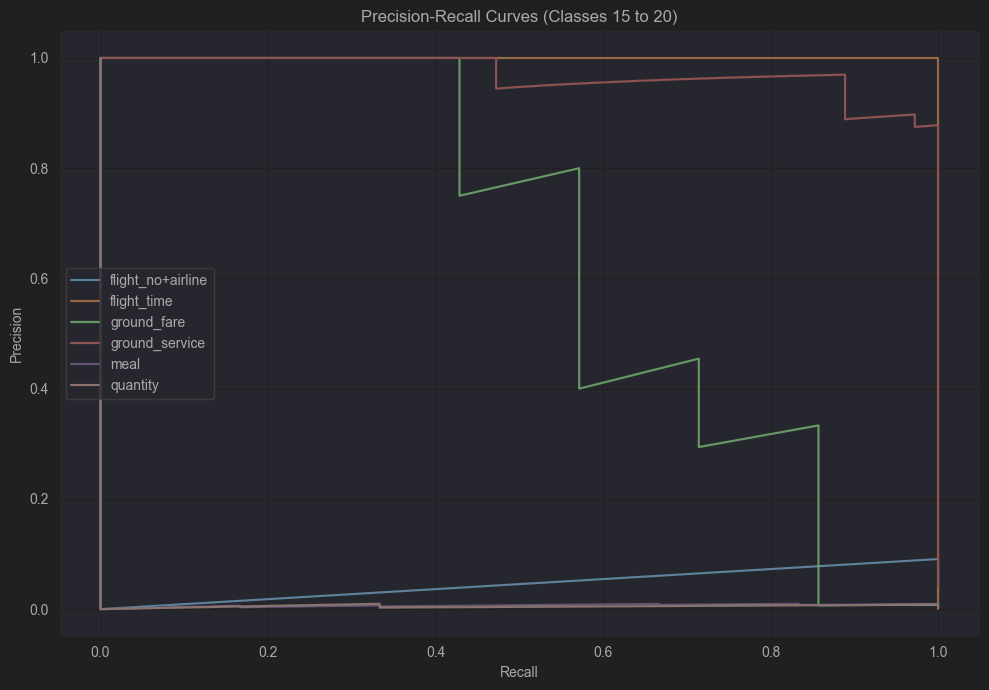
*Figure 16: Confusion Matrix of the Trigram Model*

**

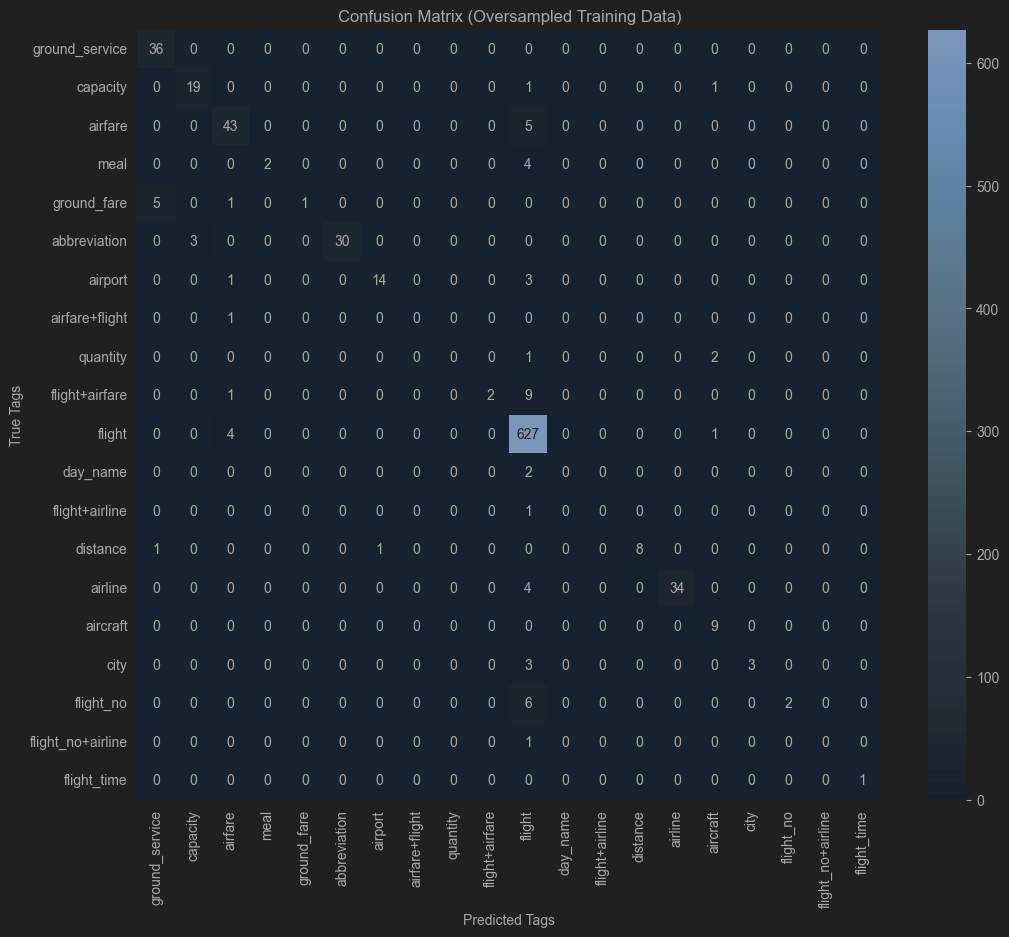
*Figure 17:* Precision-Recall curves *of the Trigram Model*

**

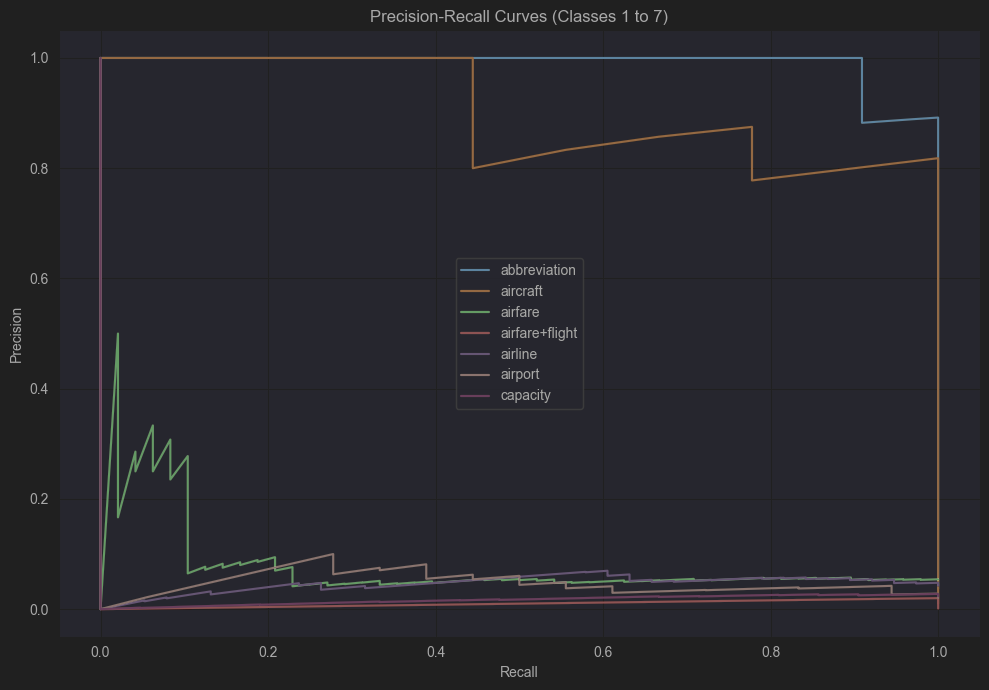
*Figure 18:* Precision-Recall curves *of the Trigram Model*

**

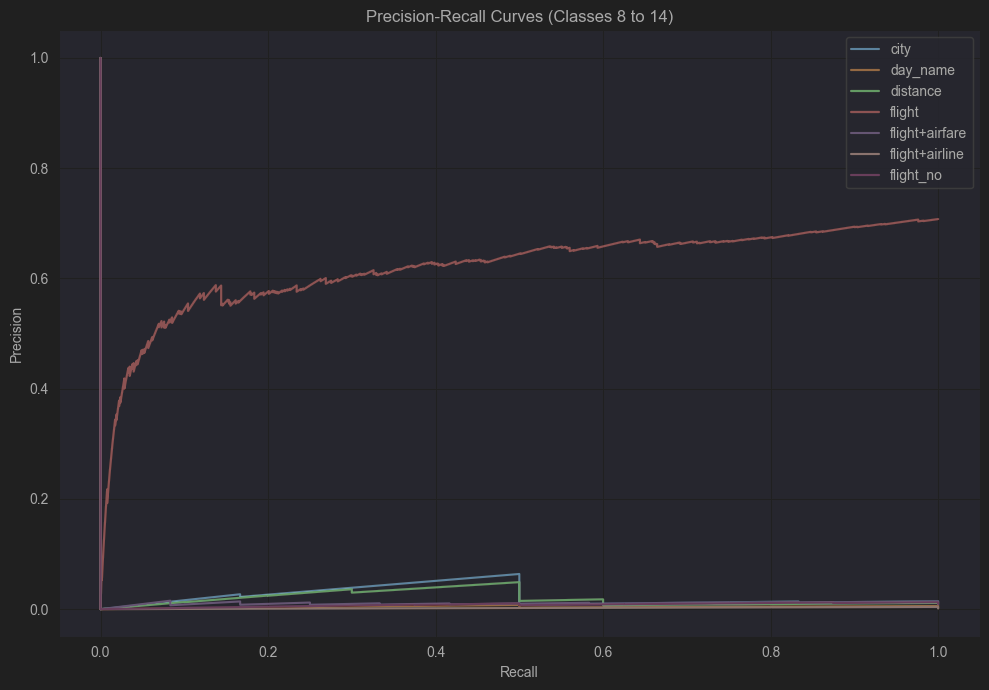
*Figure 19:* Precision-Recall curves *of the Trigram Model*



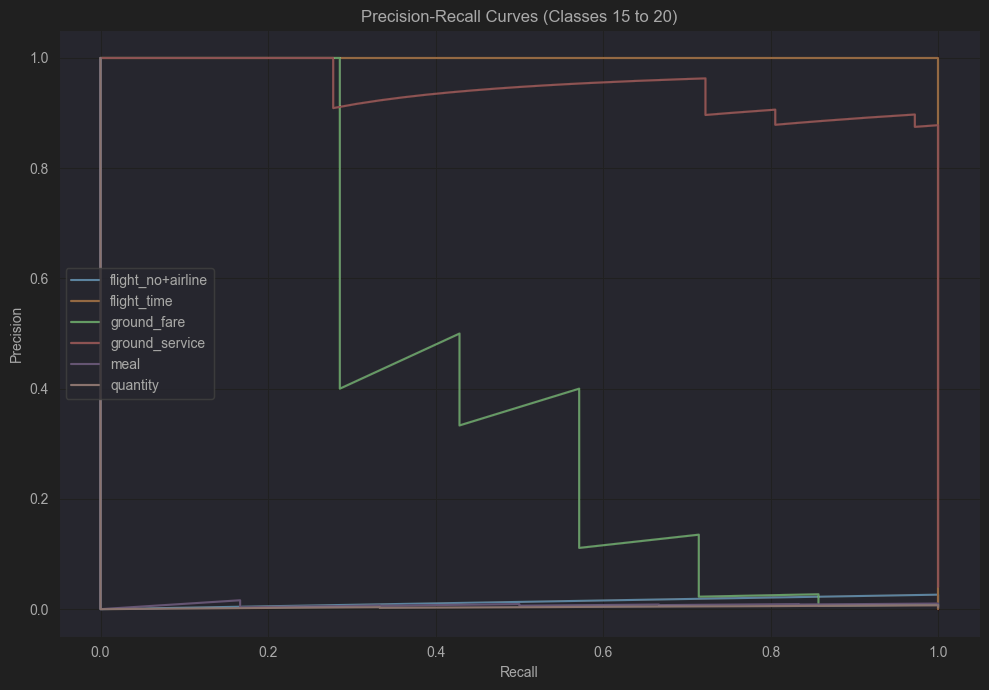
*Figure 20: Confusion Matrix of the Oversampled Bigram Model*

**

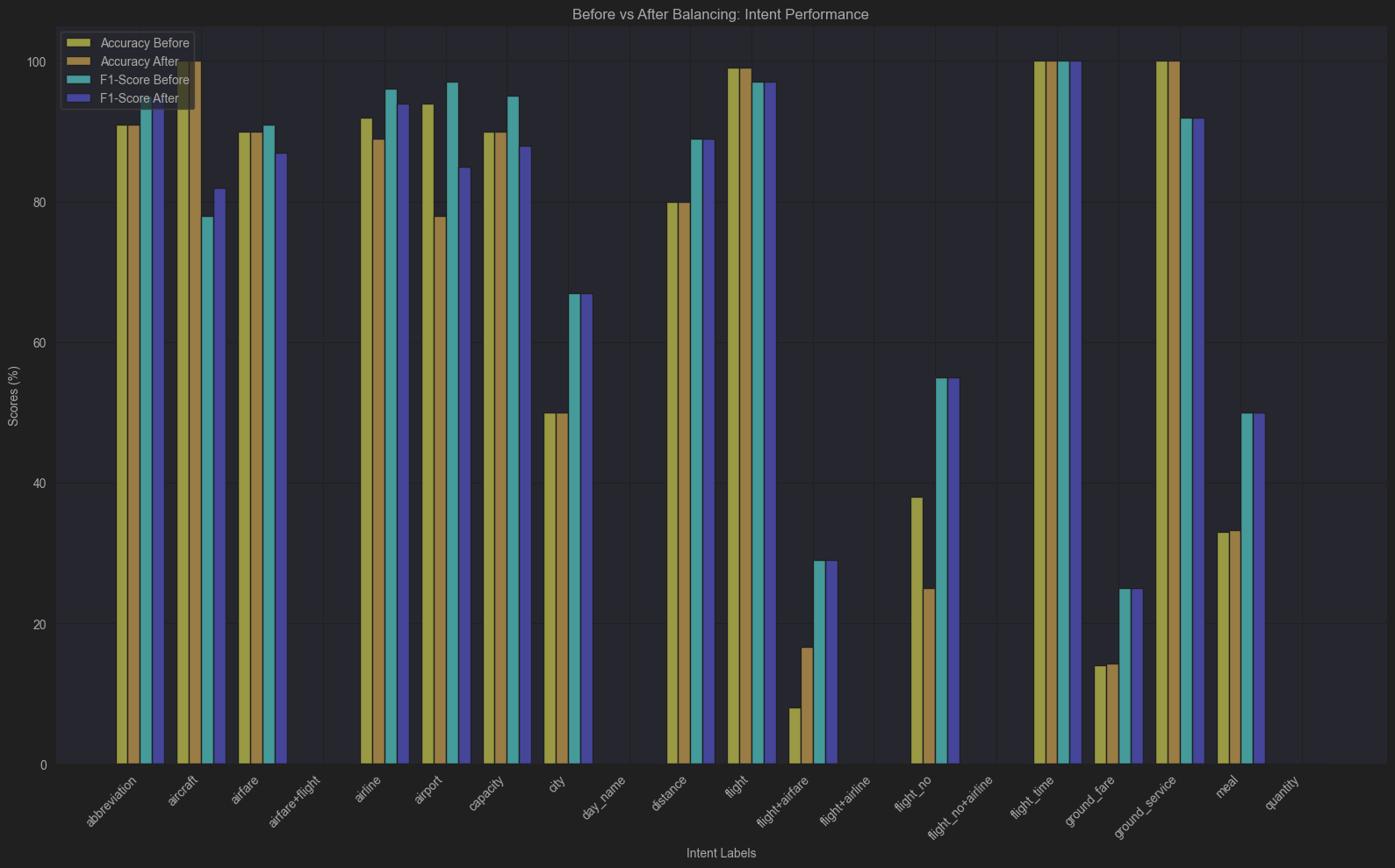
*Figure 21:* Precision-Recall curves *of the Oversampled Bigram Model*

**

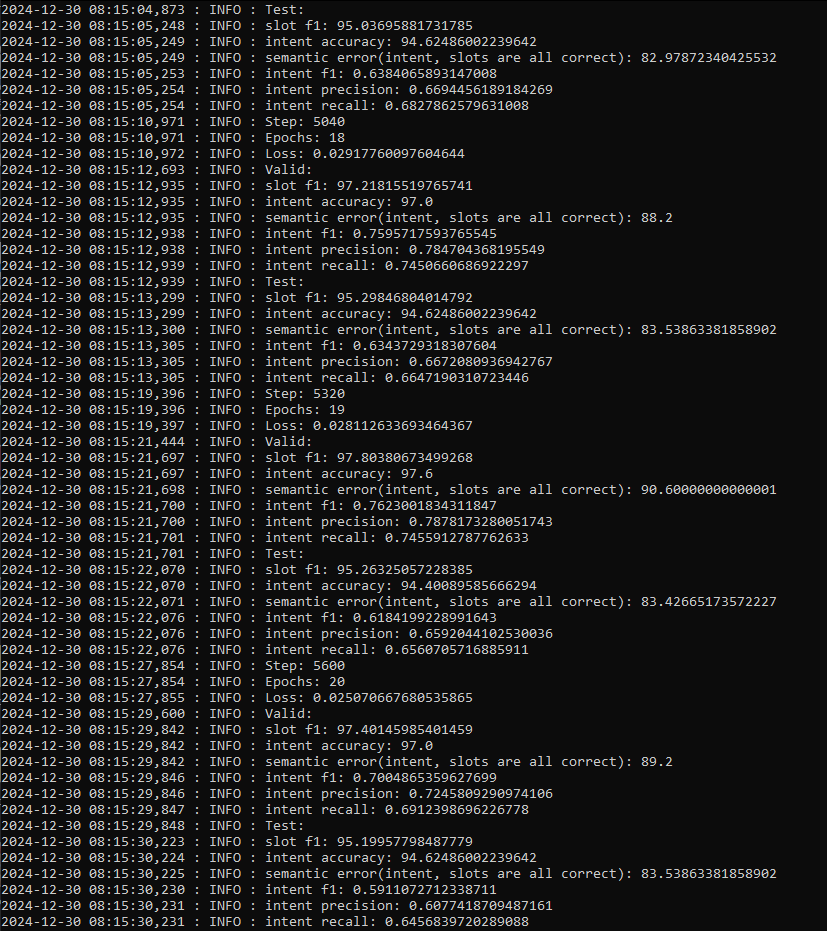
*Figure 22:* Precision-Recall curves  *of the Oversampled Bigram Model*

**

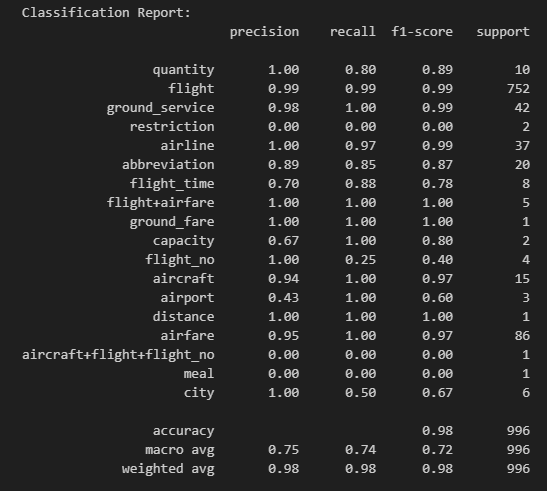
*Figure 23:*Precision-Recall curves  *of the Oversampled Bigram Model*

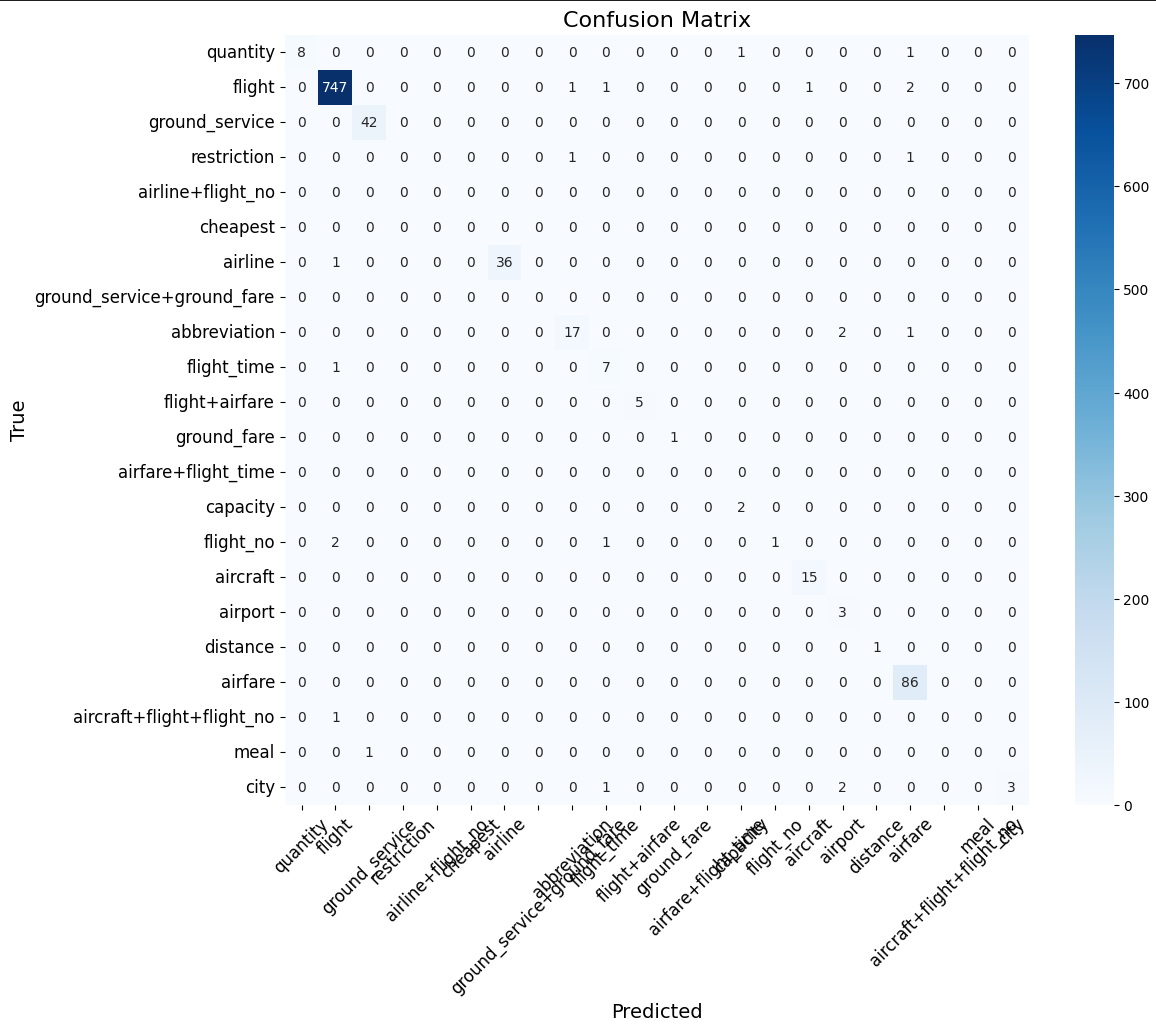


*Figure 24: With and Without Oversampling Accuracy and F1-score Comparison*

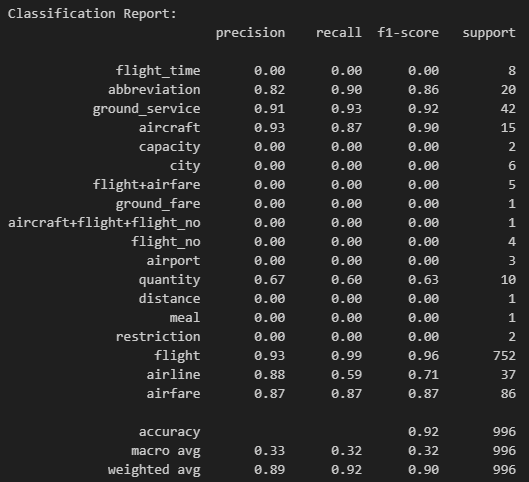


*Figure 25: Paper Implementation*

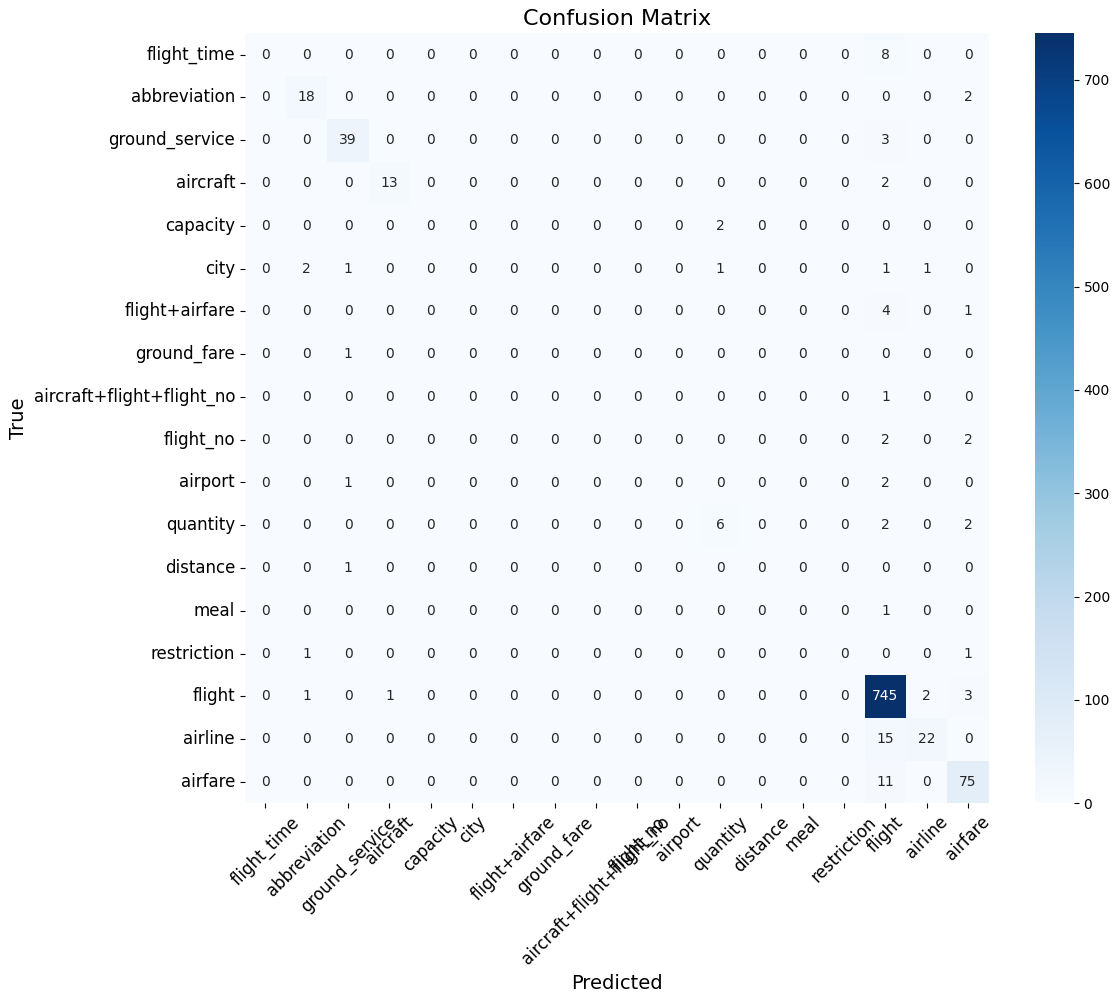
  
 *Figure 26: Neural Network with Bert Classification Report*



*Figure 27: Neural Network with Bert Confusion Matrix*

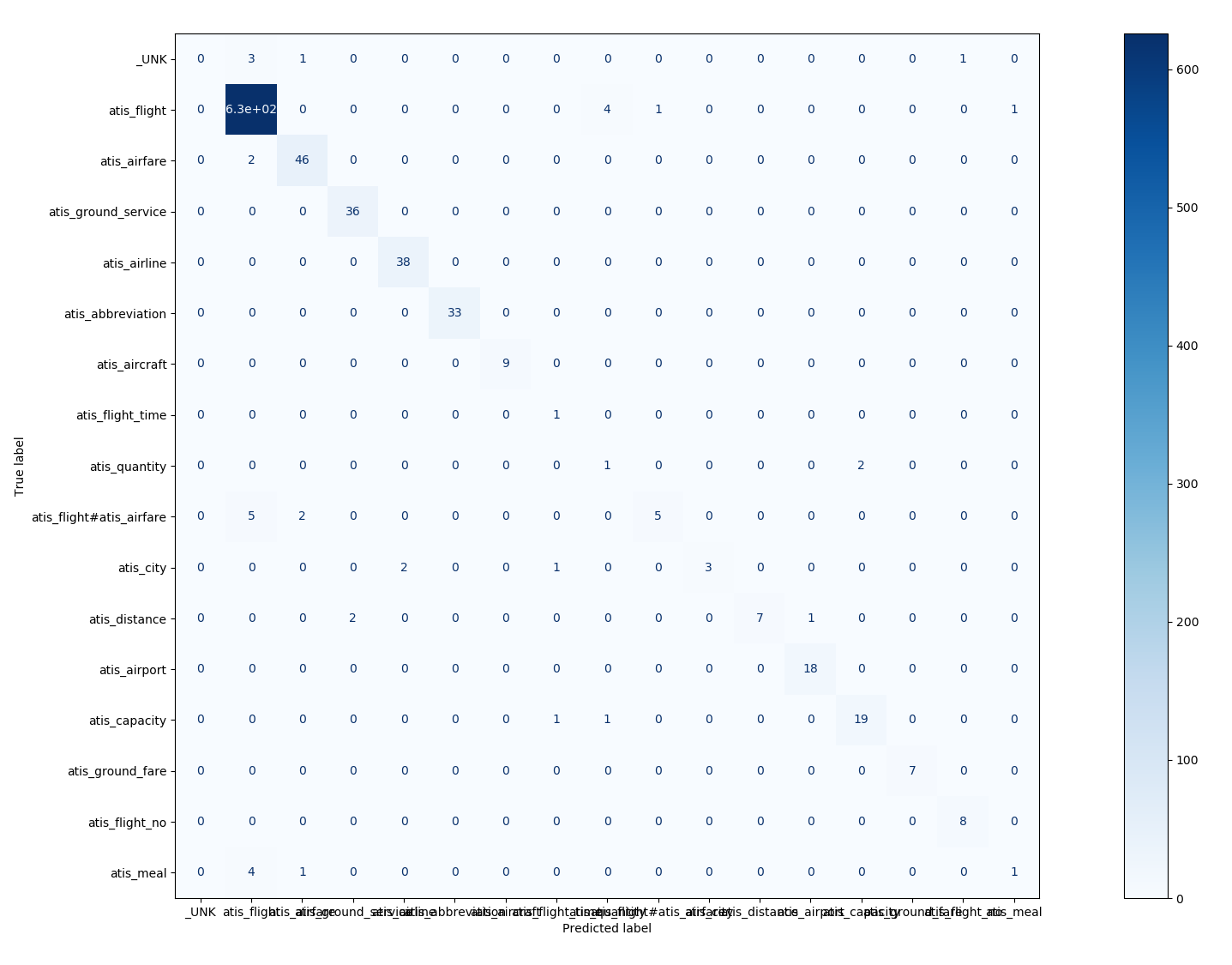


*Figure 28: Neural Network with TF-IDF Classification Report*

**

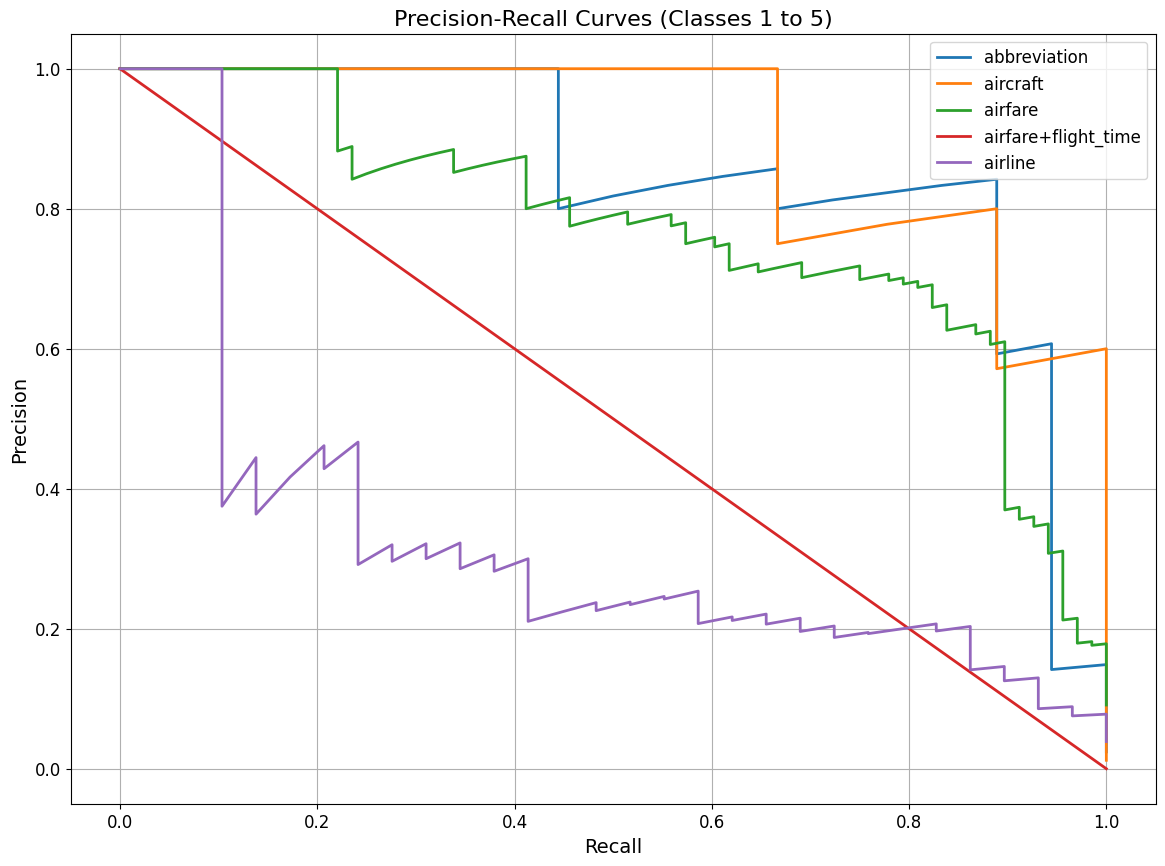
*Figure 29: Neural Network with TF-IDF Confusion Matrix*

Our model
*Figure 30: Results from improved model*

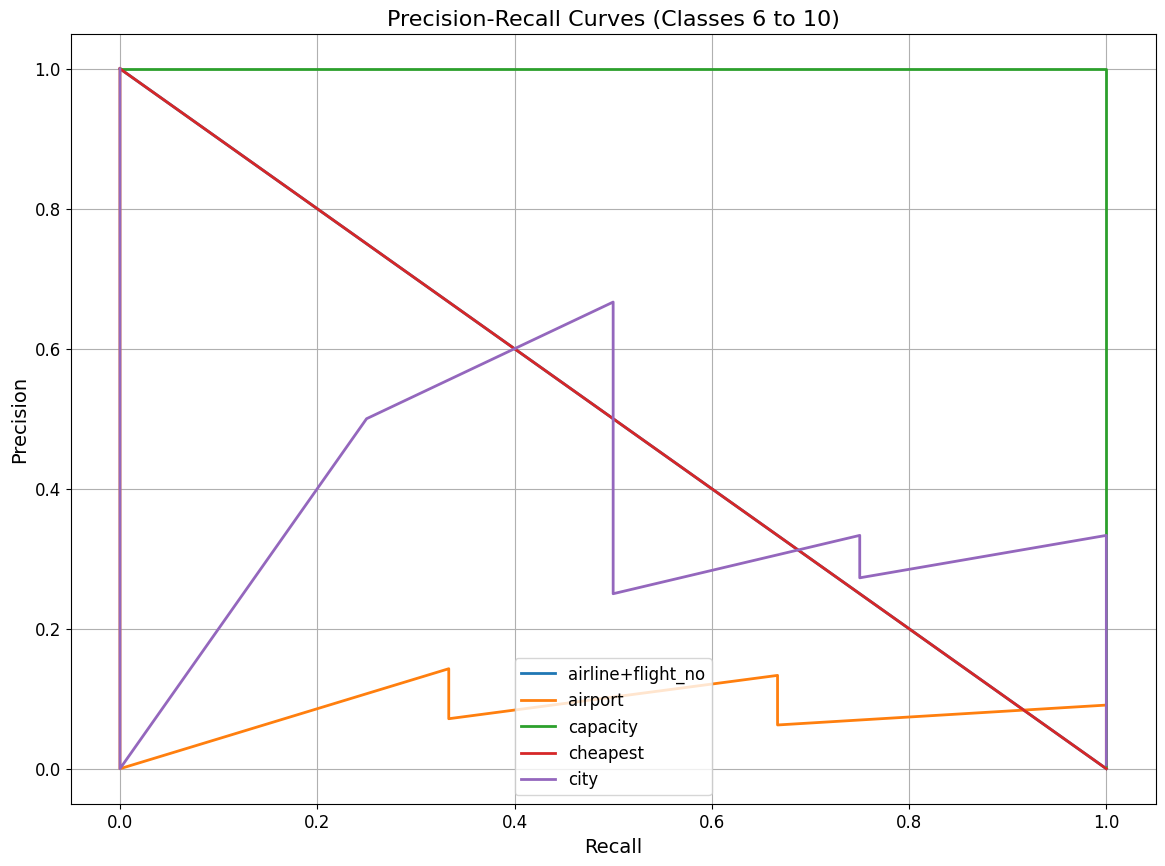


*Figure 31: Confusion matrix of the improved model*

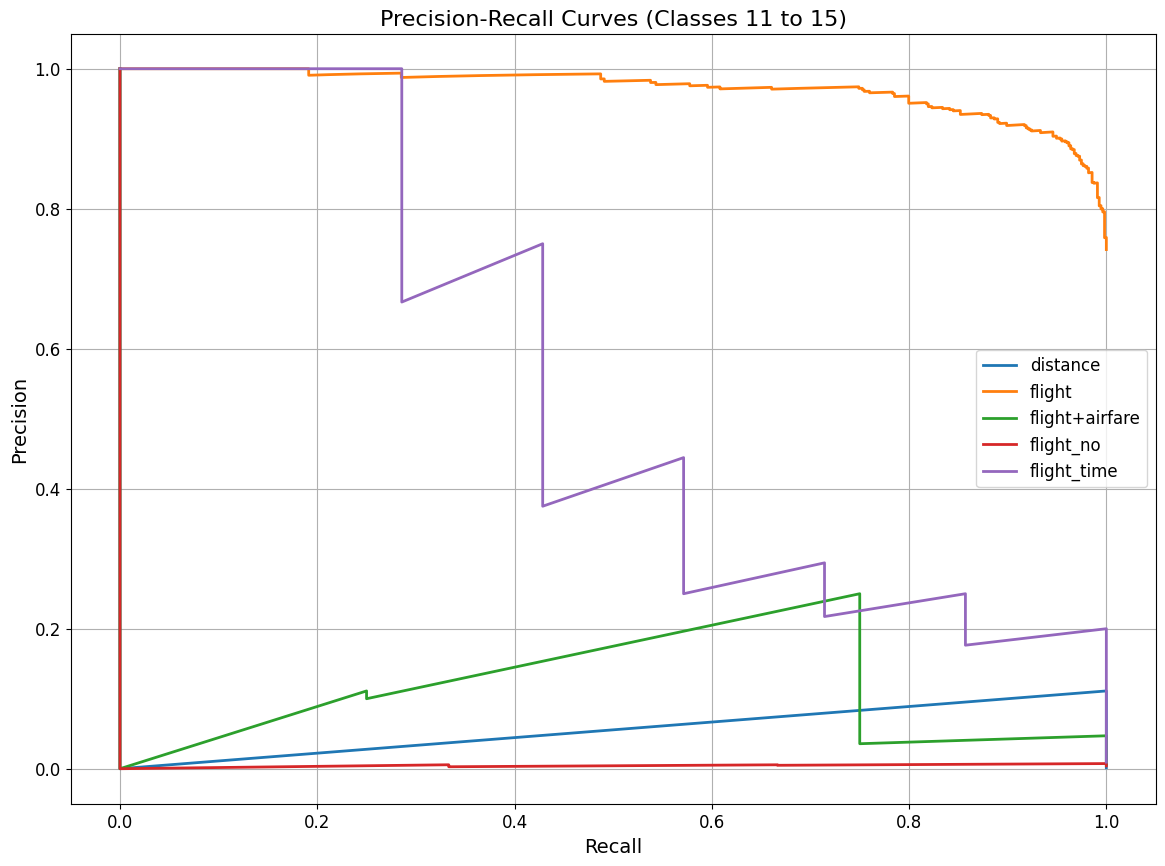
Precision-Recall curves of the baseline:



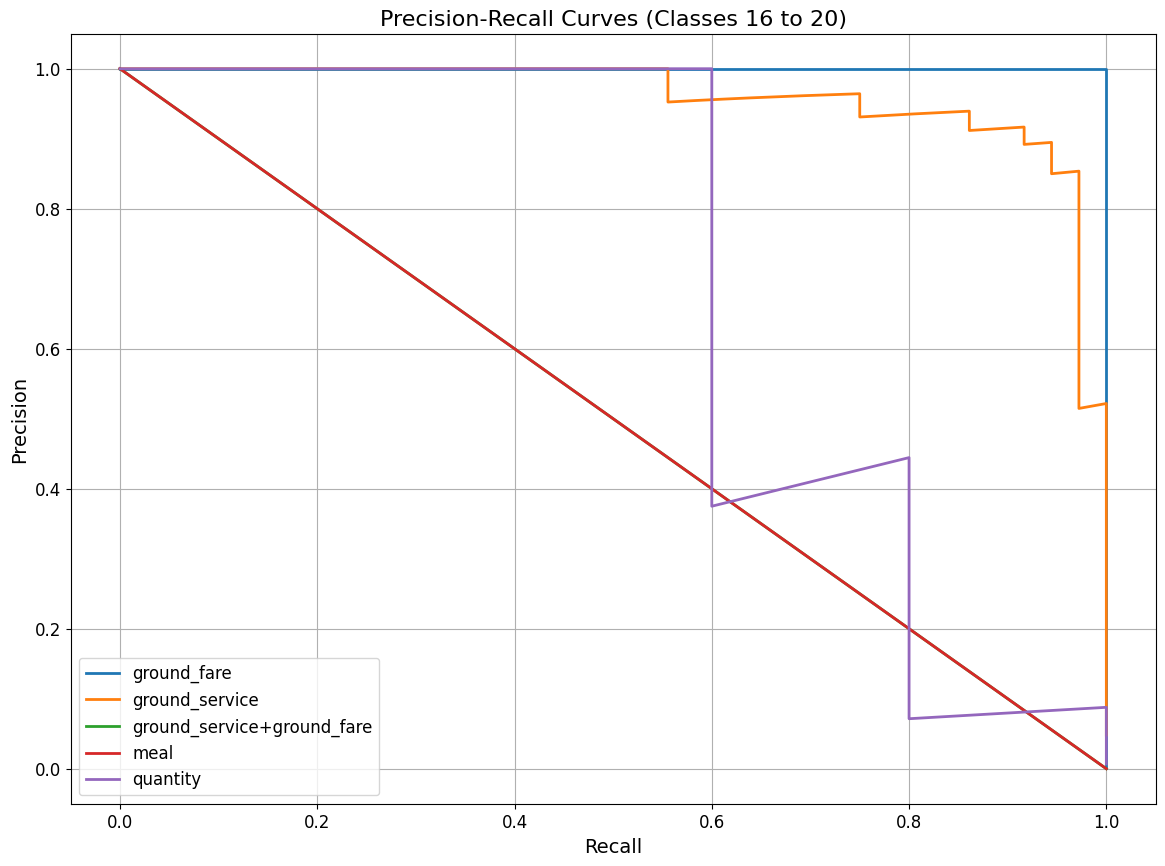
*Figure 32*



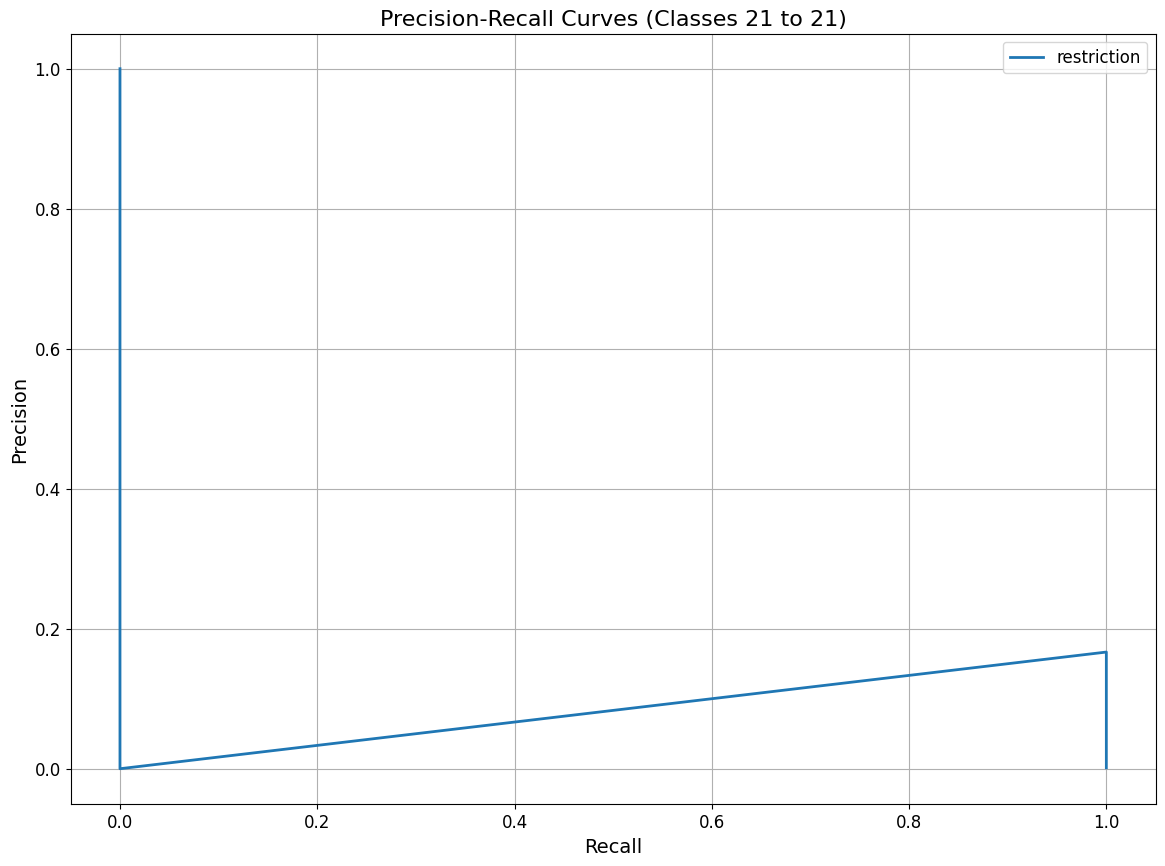
*Figure 33*



*Figure 34*

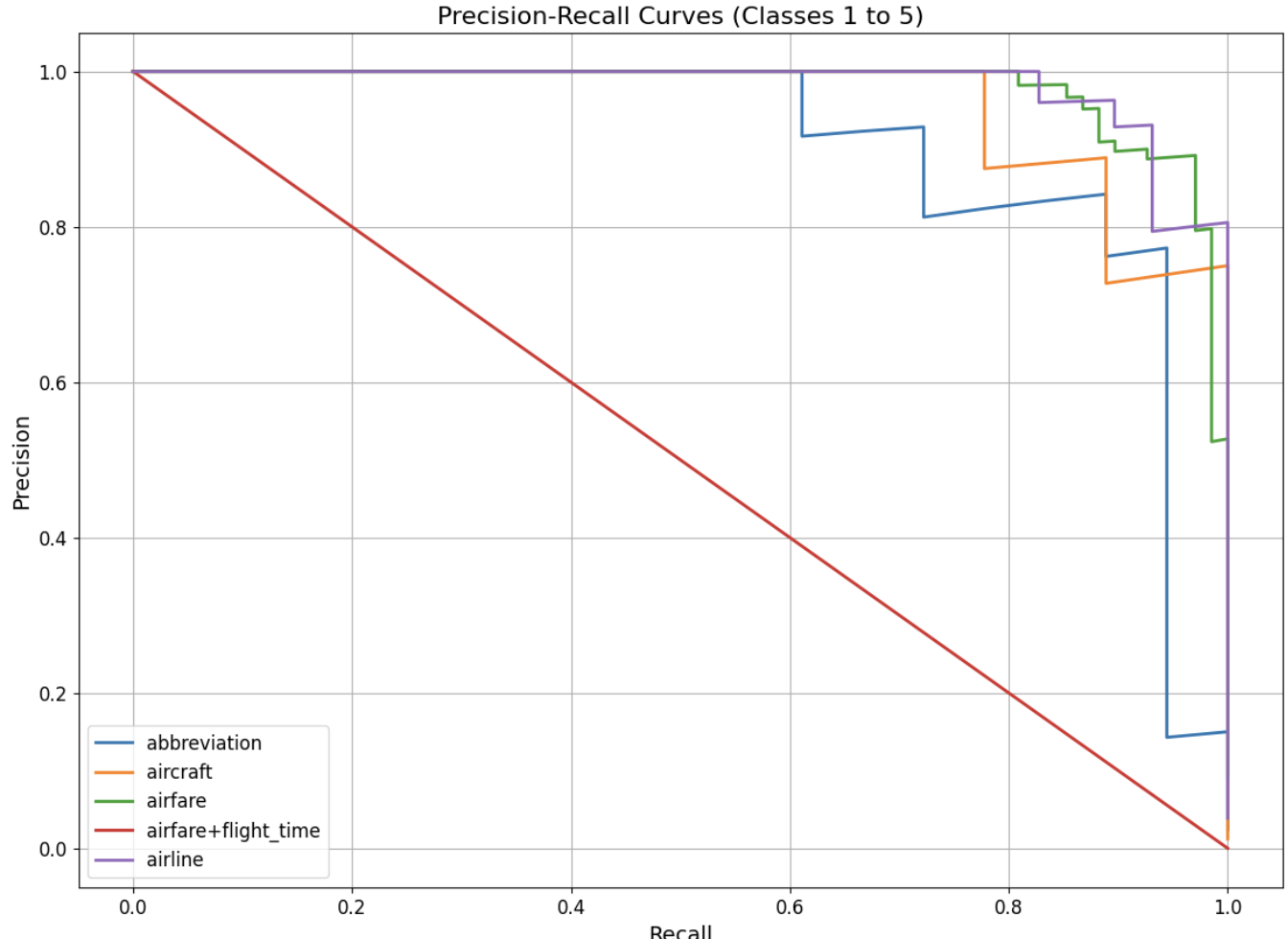


*Figure 35*

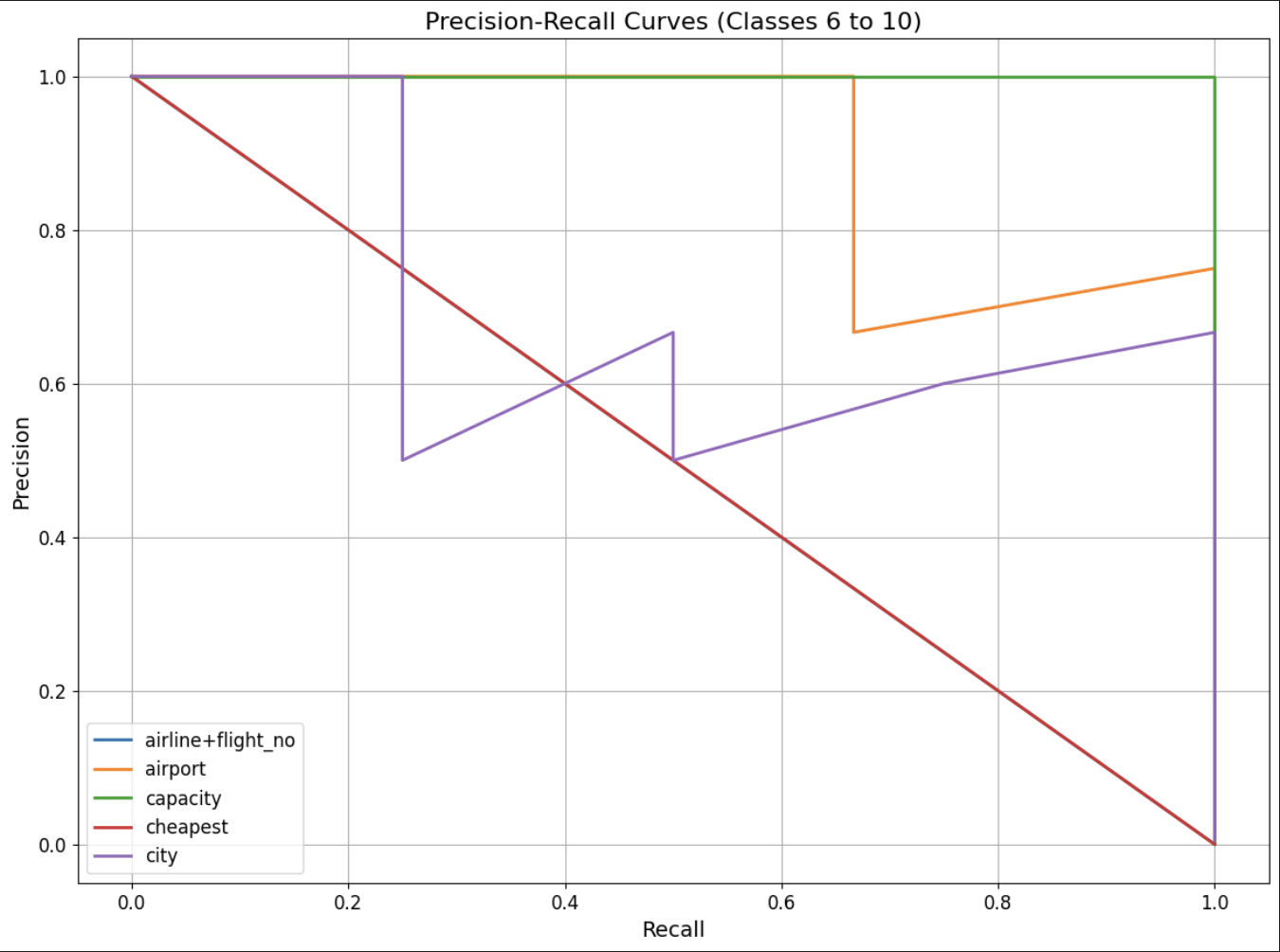


*Figure 36*

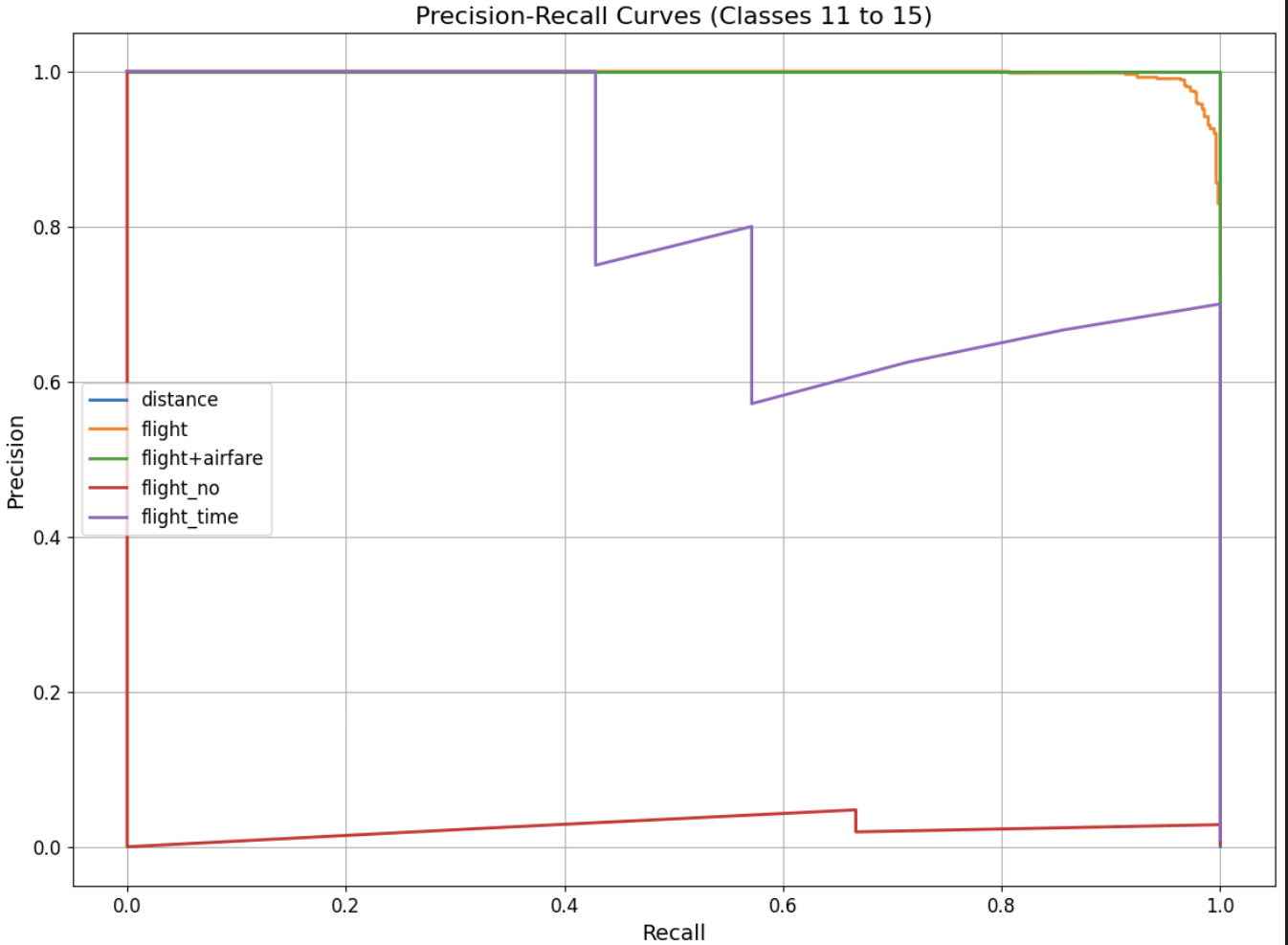
Precision-Recall curves of the Logistic Regression with TF-IDF:



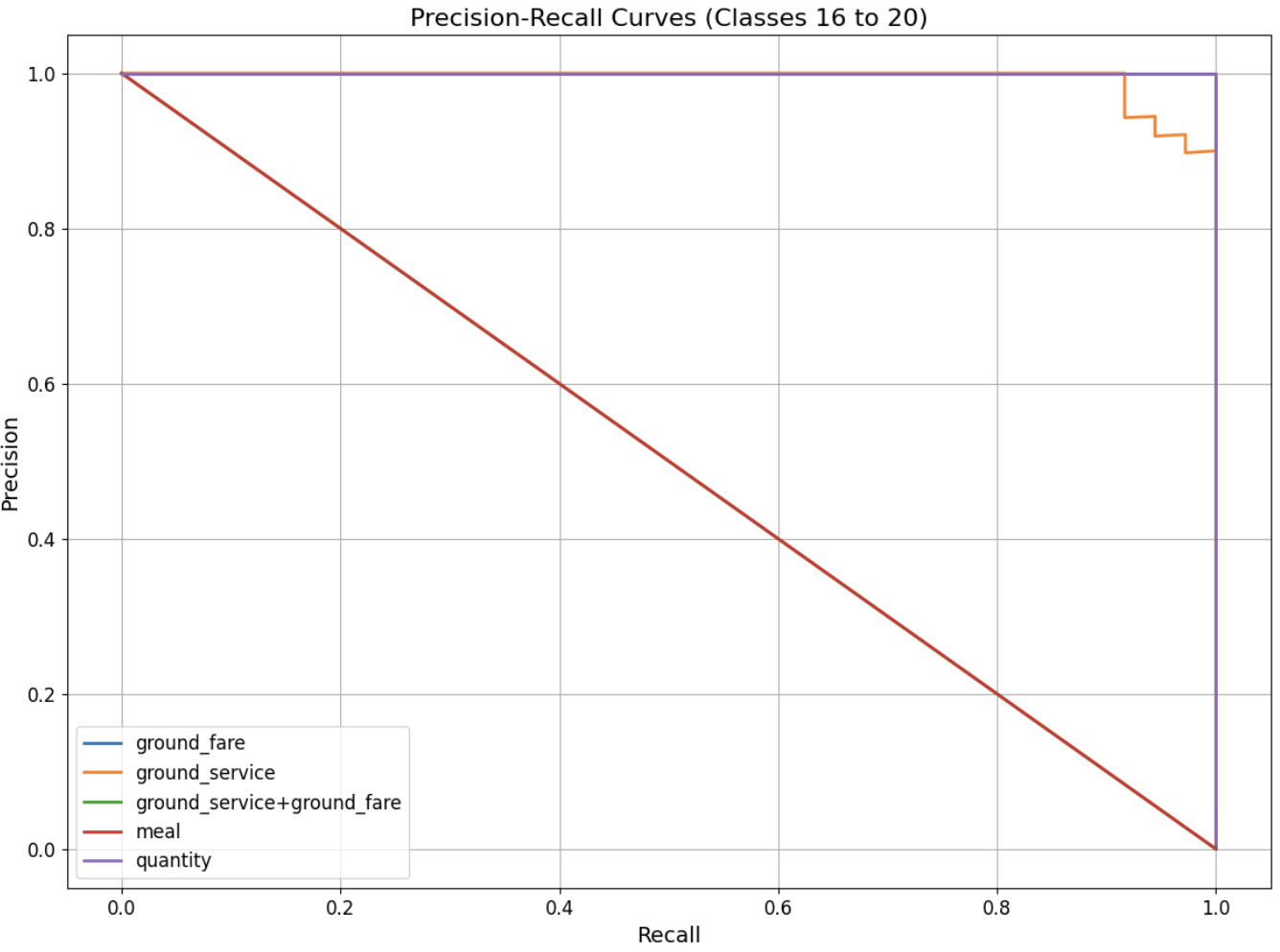
*Figure 37*



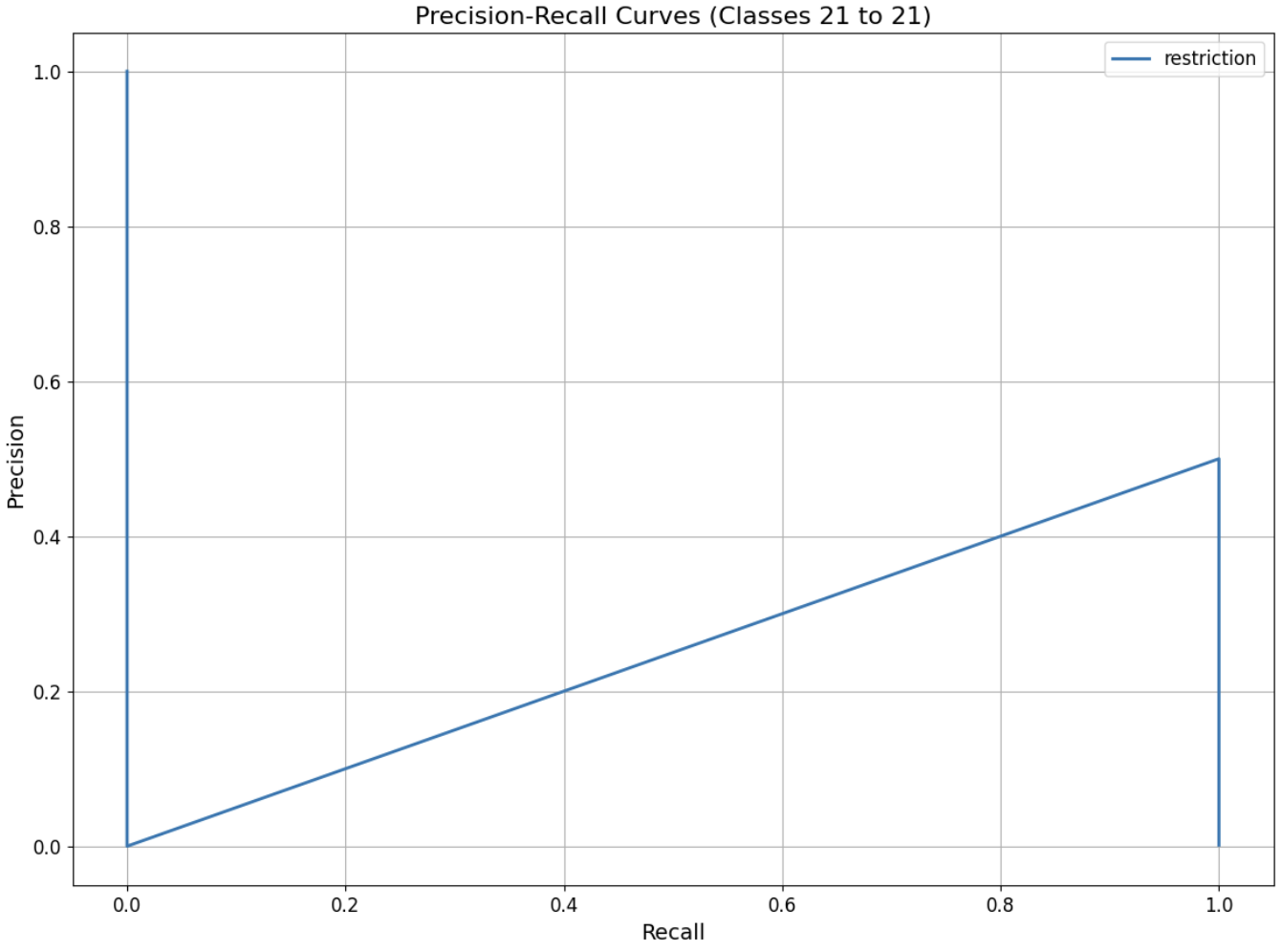
*Figure 38*



*Figure 39*

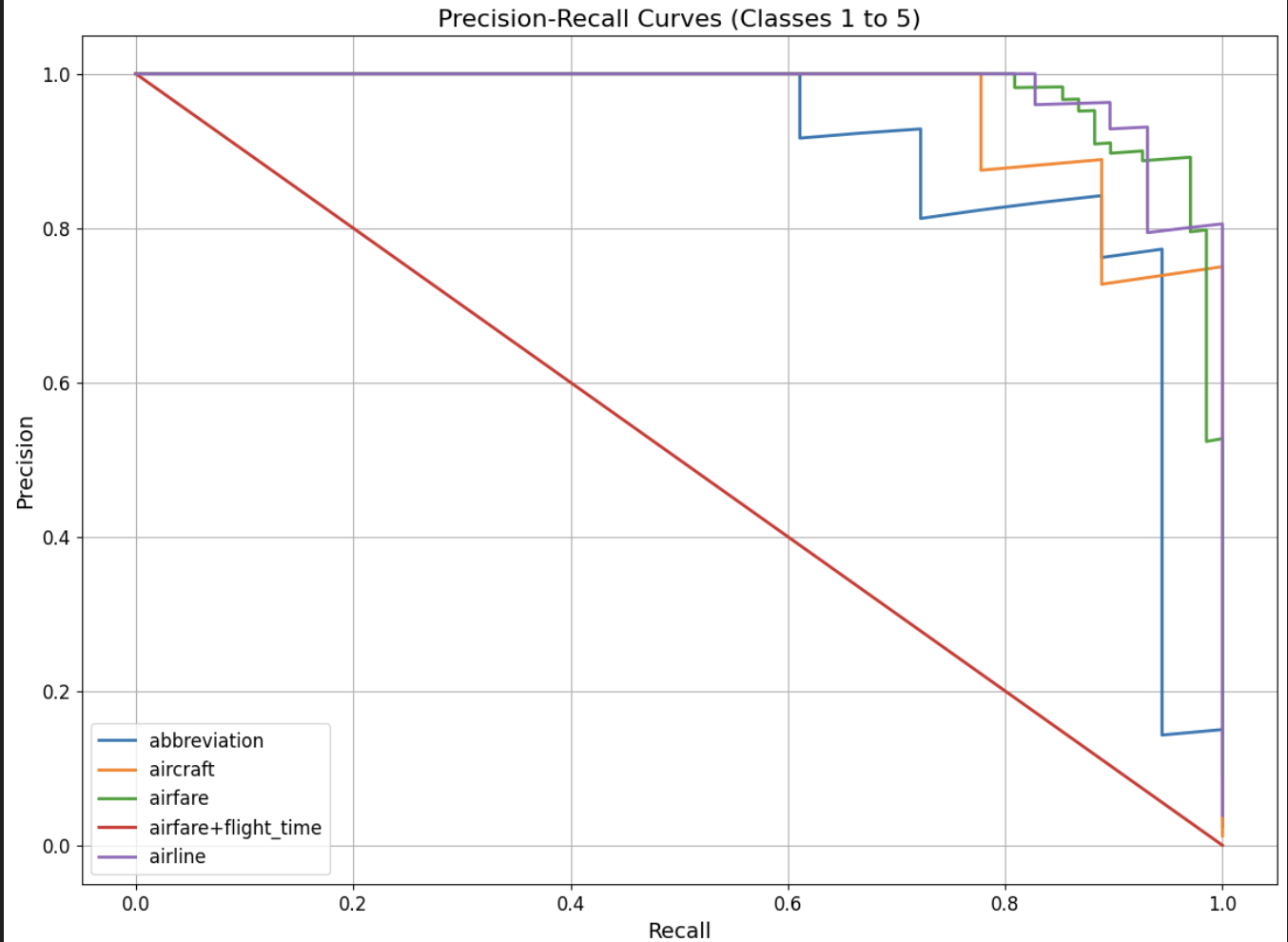


*Figure 40*

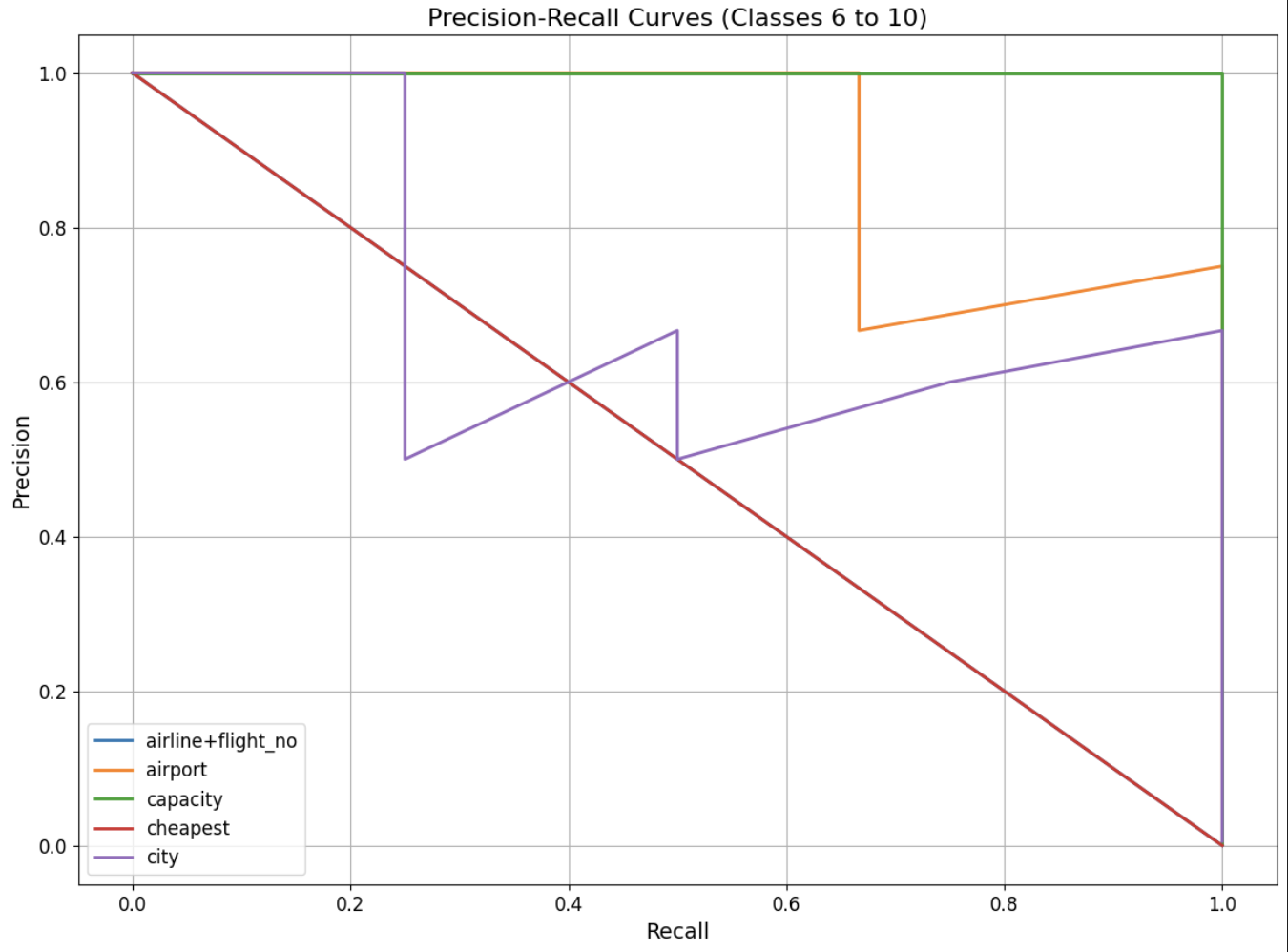


*Figure 41*

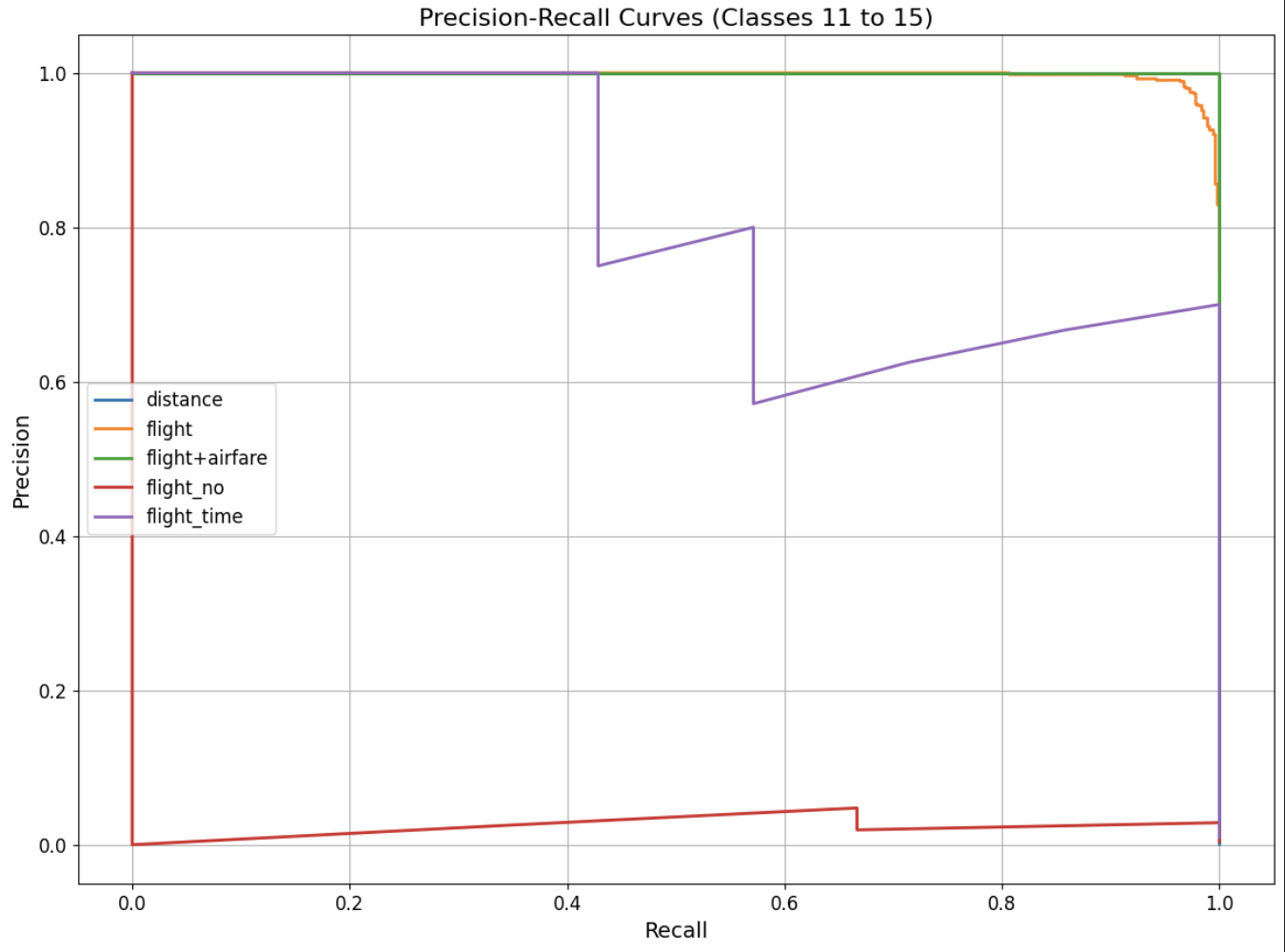
Precision-Recall curves of the Logistic Regression with Bert:



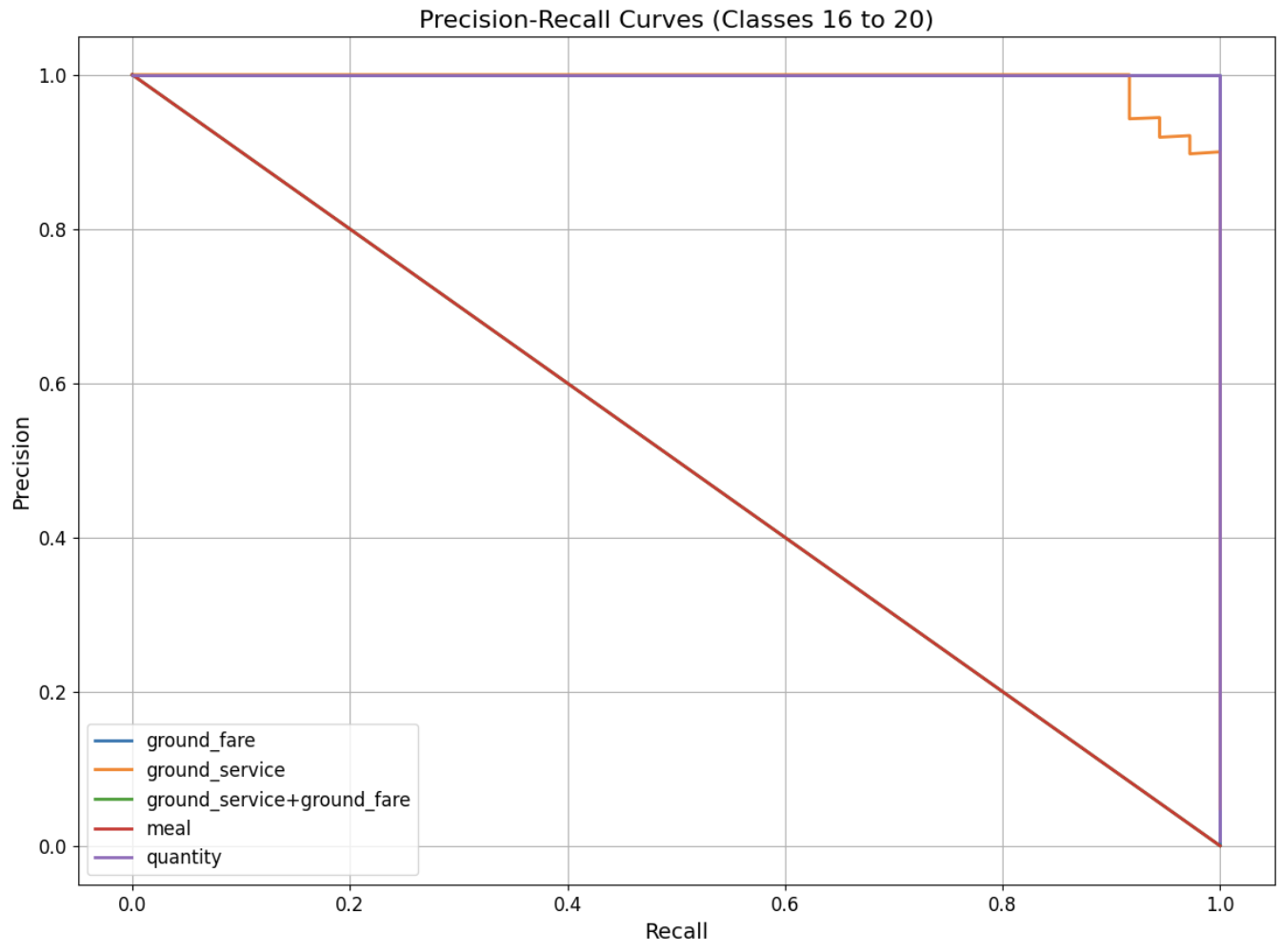
*Figure 42*



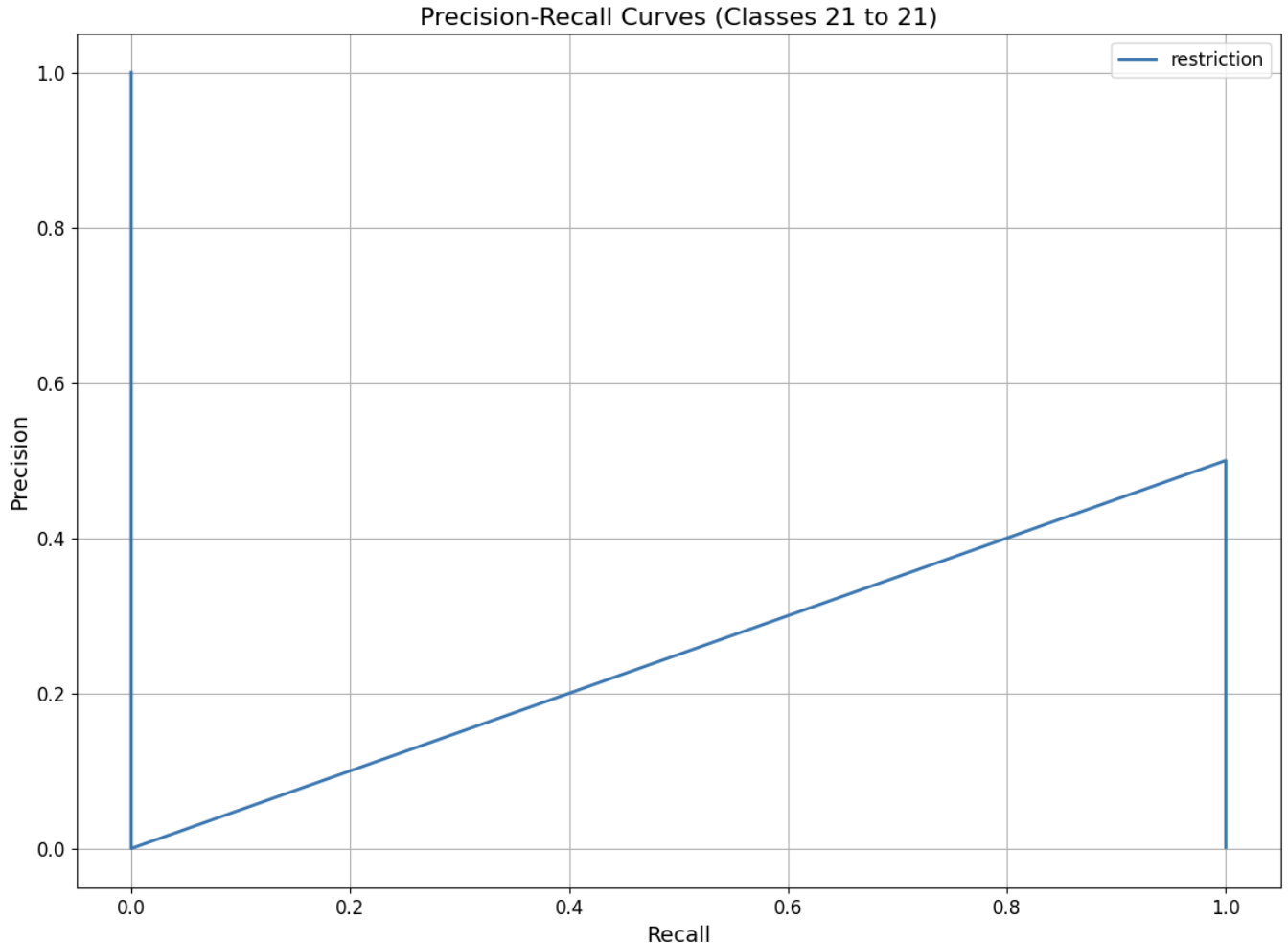
*Figure 43*



*Figure 44*



*Figure 45*



*Figure 46*