PII Masking - Mistral Take Home

Soël Megdoud

September 4, 2025

Outline

PII Masking: A Critical Business Need

What is PII Masking?

PII (Personally Identifiable Information) masking is the process of automatically detecting and anonymizing sensitive personal data in documents, ensuring privacy compliance and reducing risk.

Why PII Masking?

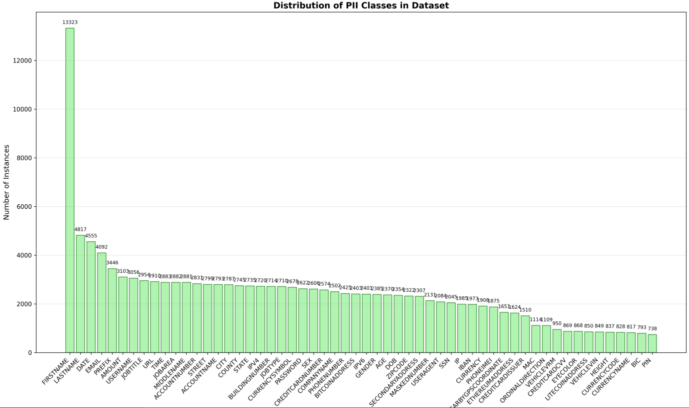
- Privacy compliance (GDPR, HIPAA, CCPA)
- Data security in pipelines
- Automated anonymization of documents

Market Opportunities

- Healthcare: Patient records
- Finance: KYC documents
- Legal: Contracts, litigation
- Enterprise: HR processing

Dataset Overview

- **Size**: 42k English + 60k French samples
- **PII Classes**: 54 comprehensive categories
- **Quality**: Human-validated synthetic data



Dataset Example

Original Text

"Hi, my name is John Smith and my email is john.smith@company.com. I live at 123 Main Street, New York, NY 10001. My phone number is (555) 123-4567."

Ground Truth Annotations

- [15:25] → FIRSTNAME: "John Smith"
- [42:65] → EMAIL: "john.smith@company.com"
- [78:94] → STREET: "123 Main Street"
- [96:104] → CITY: "New York"
- [106:108] → STATE: "NY"
- [109:114] → ZIPCODE: "10001"
- [135:149] → PHONENUMBER: "(555) 123-4567"

Mistral API Prompting Approach

Core Strategy

- **Detailed prompt**: including PII definitions with optional few-shot examples
- **Structured Output**: JSON format for entity extraction
- **Text Preservation**: No text rewriting, only entity identification
- **Position Accuracy**: Regex matching for exact positioning
- **Zero/Few-shot Learning**: Concrete examples to improve performance

JSON Output Format

```
{
  "PII": {
    "FIRSTNAME": ["John"],
    "EMAIL": ["john@company.com"],
    "STREET": ["123 Main St"],
    "PHONENUMBER": ["555-123-4567"]
  }
}
```

Evaluation Metrics & Methodology

Metrics

- Main metric: F1-score
- Additional metrics: precision/recall
- Global level and class-specific metrics versions

Strict Position Matching

- **True Positive**: Exact type + position match
- **False Positive**: Predicted but incorrect/misplaced
- **False Negative**: Missed entity
- **No Partial Credit**: Position mismatch = error
- **Token-Level Alignment**: Predictions aligned at character level, then validated against ground truth spans using exact boundary matching

Few-shot Learning & Model Comparison

Model	Without few-shot			With few-shot		
	Precision	Recall	F1	Precision	Recall	F1
Mistral Small	0.665	0.599	0.630	0.686	0.653	0.669
Mistral Medium	0.704	0.702	0.703	0.745	0.751	0.748
Mistral Large	**0.714**	**0.717**	**0.715**	**0.753**	**0.762**	**0.757**

Insight

- Few-shot learning improves results significantly
- Larger model size further boosts performance

Mistral Fine-tuning Strategy

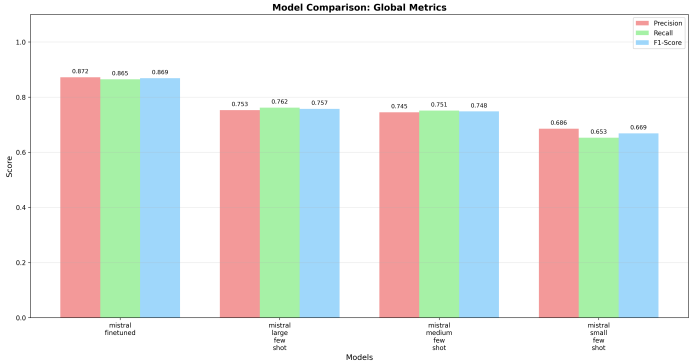
Fine-tuning Approach

- **Base Model**: Mistral-8B-latest
- **Task Format**: Same JSON structure as prompting
- **System Prompt**: Reduced for efficiency
- **Training Data**: 87k examples (FR/EN)
- **Validation**: 3k held-out examples

Hyperparameters

- Learning Rate: 1×10^{-4}
- Training Steps: 500

Fine-tuning Results & Comparison



Insight

Fine-tuning beats Mistral-Large few-shot, reaching **87%** f1-score with a much smaller model.

Performance Comparison by Classes

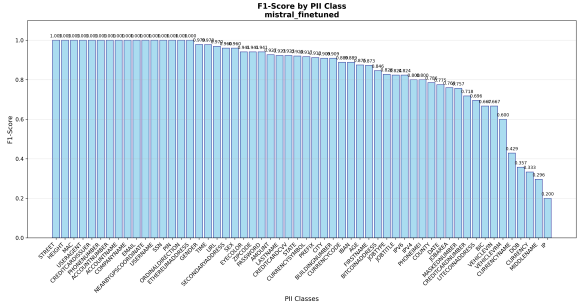


Figure: Fine-tuned MistralBaseline / Few-shot

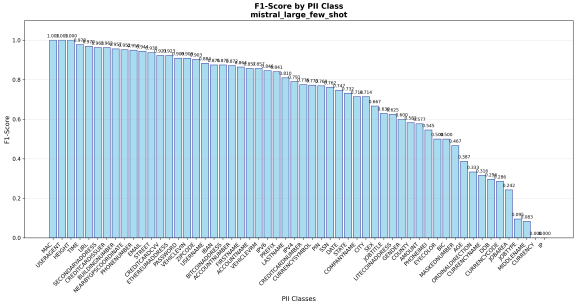


Figure: Baseline / Few-shot

Insight

Mistral fine-tuned achieves much better results on the most difficult classes. This is partly because these classes are not included as examples in the few-shot setup for Mistral Large.

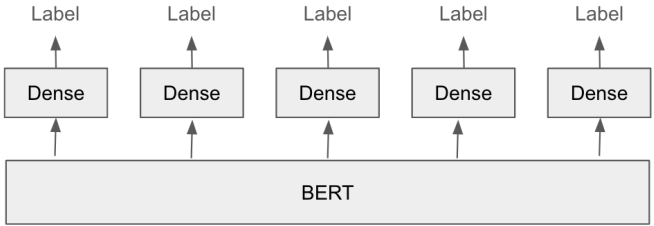
BERT Token Classification

Why Token Classification?

- **Lightweight**: BERT (100M) / DistilBERT (66M) $\ll$ LLMs
- **Bidirectional**: Full context understanding
- **CPU-Optimized**: No GPU requirements
- **Fast Inference**: $<$100ms per document
- **Specialized**: Purpose-built for NER tasks

Architecture

- Base: BERT or DistilBERT (distilled BERT-base)
- Classification head: Linear layer per token
- Subword tokenization with alignment



BERT Classifier Training

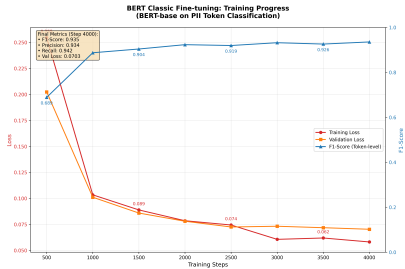


Figure: BERT Training Loss

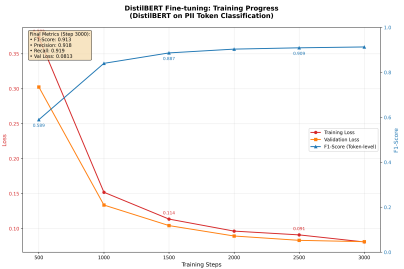
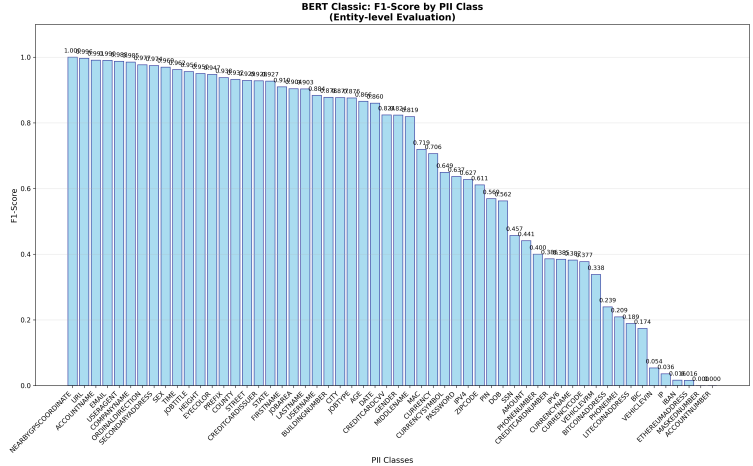


Figure: DistilBERT Training Loss

Insight

Both models reach ∼90% token-level F1-score in only 3 epochs!

BERT Results by Classes



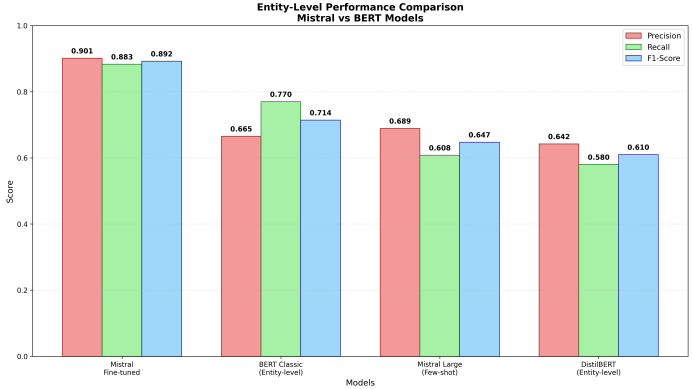
Entity-Level Results

- **BERT**: 71.4% F1
- **DistilBERT**: 61% F1
- Gap: 20–30% performance drop!

Why the Gap Exists

- Boundary detection errors
- Single token error = entire entity failed

Overall Benchmark



Insight

BERT non distilled beats Mistral Large with few-shot but Mistral fine-tuned remains the best across all metrics.

Token Classification with LLM

Approach

- Base: **Mistral-8B** used as encoder
- Extract latest latent representation for each token
- Backbone weights frozen, fine-tune only classification head
- Classification head trained to predict token classes

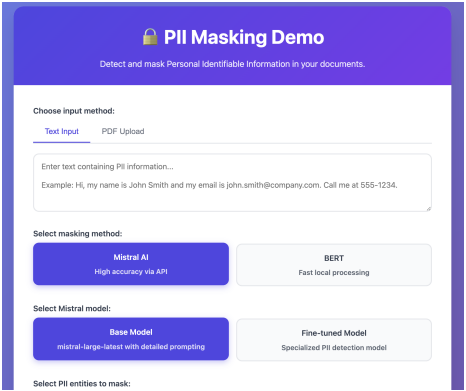
Results

- Trained for **1 epoch** due to limited time and GPU resources
- Achieved $\sim$85% token-level F1-score (similar to BERT baselines)
- Did not push further testing but the code support is in the repo

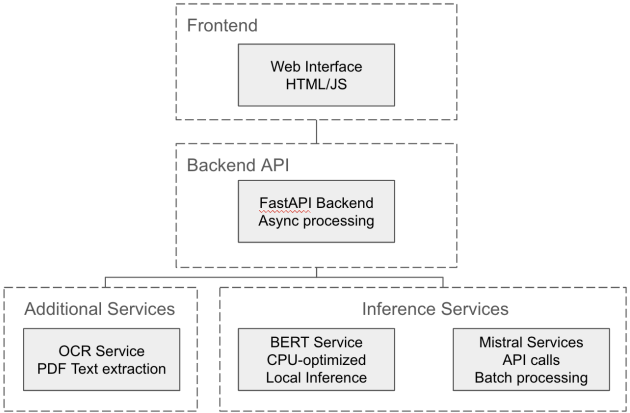
Containerized Solution & Deployment

HuggingFace Space Deployment

- **Containerized Application**: Using Docker container
- **Live Demo**: hugging-face.co/spaces/SoelMgd/pii_masking
- **Multi-method Support**: All 3 approaches available
- **PDF Processing**: Drag-and-drop with OCR
- **Entity Selection**: Granular PII type control
- **Real-time Processing**: WebSocket updates



System Architecture



Component Overview

- **Frontend**: HTML/JS
- **FastAPI Backend**: Async processing with different inference services
- **BERT Service**: CPU-optimized local inference (HF CPU)
- **Mistral Services**: Chunking and batch processing to speed up long text inference
- **OCR Service**: Mistral OCR for PDF text extraction

Efficient Async/Sync Architecture

The challenge

- **BERT**: CPU-bound, blocking ops
- **Mistral**: I/O-bound, async-ready
- **FastAPI**: Async framework
- Goal: Serve multiple users simultaneously

Bottlenecks of naive approach

- BERT blocks event loop
- Sequential processing
- Poor scalability

Proposed solution

- Unified `await` interface for all services
- **Mistral**: native async calls
- **BERT**: offloaded to thread pool

Benefits

- Non-blocking BERT inference
- Efficient Mistral calls
- Scales to multiple concurrent requests

Architecture Validation: Concurrency Benchmark Results

Experimental Setup

- 5 simultaneous requests with 500 characters
- Sequential vs Concurrent processing comparison
- Environment: 8-core CPU, 8GB RAM, Python 3.12

Service	Sequential	Concurrent	Speedup
BERT (async/sync)	1.37s	0.23s	**6.0x faster**
Mistral Large API	8.77s	8.85s	Same

Insight

Async/sync architecture delivers 6x concurrency speedup for BERT. BERT is also **25x faster** in inference time than LLMs.

Business Impact: ROI

Automated PII Detection vs Manual Review

Method	Cost Structure	100k Documents	Cost per Doc
Mistral L + Review	\$3.00/1M tokens + 10s review	\$11,126	\$0.11
Mistral F + Review	\$0.10/1M tokens + 10s review	\$11,112	\$0.11
Manual Review	\$40/hour (30 sec review)	\$33,333	\$0.33

Hypothesis:

- 10 PIIs per doc, 5 output tokens per PII + 10s human reviewing at \$40/hour
- Manual processing: 30 seconds per document at \$40/hour

ROI Analysis

Automation with human review delivers 66.7% cost reduction vs full manual processing. The solution provides 3x faster processing while maintaining quality control through human oversight.

Key Findings & Recommendations

Performance Summary

- **Mistral Fine-tuned**: 89.2% F1
- **BERT**: 71.4% F1, 25x faster
- **Mistral Large Few-shot**: 64.7% F1
- **DistilBERT**: 61% F1

Technical Achievements

- Multi-method comparison
- Comprehensive evaluation framework repository
- Production-ready deployment
- Efficient async/sync architecture for concurrency

Business Recommendations

- Max performance: Mistral Fine-tuned
- Best performance/speed: BERT fine-tuned

Codebase Structure & Extensibility

Modular Framework

- `src/pii_masking/`: Core library
- `experiments/`: Research implementations
- `space/`: Production deployment

Key Design Principles

- Abstract base classes for extensibility
- Standardized evaluation pipeline accros approaches
- Configuration-driven experiments
- Reproducible results with export and visualization support

Thank You

Ressources

Live Demo Available:
huggingface.co/spaces/SoelMgd/pii_masking

Repository:
github.com/SoelMgd/PII$_{m}$*asking*