Food Safety Review Classifier (Chinese)
Data Mining

Sangeen Khan

December 31, 2025

1 Abstract

Food safety complaint detection from user reviews is a high impact text classification task because missed positives can hide genuine hygiene risks, while false alarms can create unnecessary noise for downstream moderation. This report presents a GPU friendly Chinese text classification pipeline optimized for an F1 based competition setting. The system uses a strong Chinese transformer backbone (MacBERT), stratified 5 fold cross validation, and out of fold (OOF) probability aggregation to select a decision threshold that maximizes the positive class F1 score. To improve signal quality without heavy feature engineering, each review is lightly normalized and augmented with a short task prompt and a keyword based tag that indicates whether food safety risk terms are present. The report documents the training design, the threshold search strategy, and a set of diagnostic plots, including F1 versus threshold, ROC, Precision Recall, confusion matrix, calibration curve, and training curves. Using OOF predictions, the system achieves high discrimination (ROC AUC 0.9911, AP 0.9633) and strong overall classification performance (accuracy 0.9766, F1 0.9196) with a confusion matrix of $\begin{bmatrix} 8427 & 62 \\ 172 & 1339 \end{bmatrix}$. These results indicate that the backbone model separates classes well, and that the remaining improvements are mainly related to threshold stability, calibration, and distribution shift between local validation and the hidden test set.

2 Introduction

2.1 Problem Context

Online food delivery and restaurant platforms generate large volumes of user reviews. Within these reviews, a subset contains complaints that can indicate hygiene and safety risks, such as spoiled food, foreign objects, undercooked dishes, or symptoms like vomiting and diarrhea. Detecting such complaints automatically supports content moderation, early warning systems, and quality monitoring. The challenge is that safety complaints are usually a minority class and can be expressed with informal language, short texts, emojis, and slang. This makes the task sensitive to class imbalance and hard negative examples.

2.2 Why F1 Matters

In this competition, performance is evaluated using the F1 score for the positive class. Unlike accuracy, F1 explicitly balances precision and recall. Precision is penalized by false positives, while recall is penalized by false negatives. In practice, missing true safety complaints can be costly, while too many false positives can overwhelm manual review. Because F1 depends on the decision threshold applied to predicted probabilities, selecting a threshold that is consistent with the target metric is critical.

2.3 Main Goals of This Work

This work aims to build a system that is:

- **Accurate under F1 scoring**: Explicitly tune the decision threshold using OOF predictions rather than relying on a default 0.5 threshold.
- **Stable and low variance**: Use stratified cross validation to reduce sensitivity to a single train validation split.
- **Efficient on limited hardware**: Run reliably on a Colab GPU with 12 GB VRAM using FP16 and gradient accumulation.
- **Interpretable through diagnostics**: Provide plots that explain discrimination, calibration, training behavior, and threshold tradeoffs.

2.4 Contributions

The report makes the following contributions:

- A reproducible cross validated transformer pipeline for Chinese food safety complaint detection.
- OOF based threshold selection aligned with the positive class F1 metric.
- A lightweight prompt and keyword tag injection that improves sensitivity to safety related language with minimal complexity.
- A complete diagnostic suite with plots that support error analysis and threshold decisions.

3 Methodology

3.1 Data Interface and Assumptions

The system assumes the following file formats:

- `train.csv`: two fields, label and comment. The label is binary, where 1 indicates a food safety or hygiene complaint.
- `test.csv`: two fields, id and comment. Labels are unknown for the test set.

The code includes robust path resolution and attempts both tab separated and comma separated reading for training data to handle common formatting issues.

3.2 Text Normalization

Each review text is normalized using a minimal transformation to avoid removing important signals:

- Stripping leading and trailing whitespace.
- Collapsing repeated spaces.
- Normalizing common repeated punctuation patterns into a compact form.

This step is intentionally lightweight. Aggressive cleaning can remove cues that correlate with complaint language, such as repeated punctuation or short emotional expressions.

3.3 Task Prompting and Keyword Tagging

To guide the model toward the specific decision boundary required by the task, the input is augmented with:

- A **task prompt** that explicitly defines the classification objective.
- A **binary keyword tag** that signals whether the review contains any risk related terms.

Let x be the normalized review. We define a keyword set $\mathcal{K}$ containing terms such as *vomiting*, *diarrhea*, *spoiled*, *mold*, *foreign object*, *undercooked*, and *dirty*. We compute:

$$\mathrm{hit}(x) = \mathbb{I}\left[\exists k \in \mathcal{K} \text{ such that } k \in x\right].$$

Then the final model input is:

$$x' = \mathrm{prompt} \mid\mid \mathrm{tag}(\mathrm{hit}(x)) \mid\mid x,$$

where the tag is `[Food Safety Clues]` if there is a hit, otherwise `[No Obvious Safety Clues]`.

These tags are added as special tokens to the tokenizer vocabulary, and the model embedding matrix is resized accordingly. This lets the model learn a distinct representation for the tag without relying on subword fragmentation.

3.4 Backbone Model: Chinese MacBERT-base

The core classifier in this system is `hfl/chinese-macbert-base`, a transformer encoder model in the BERT family. We fine-tune a Chinese MacBERT base model as the single backbone, using the Hugging Face checkpoint `hfl/chinese-macbert-base`. This backbone is an encoder-only Transformer with a BERT-base configuration: (i) $L = 12$ Transformer encoder layers, (ii) hidden size $H = 768$, (iii) $A = 12$ self-attention heads, and (iv) approximately 102M parameters. For our task, we attach a 2-class classification head on top of the final `[CLS]` representation and optimize the model end-to-end for binary food-safety complaint detection. During fine-tuning, inputs are truncated to a maximum sequence length of 192 tokens.

Given an input sequence of tokens $x' = (t_1, \ldots, t_L)$, the model maps tokens to embeddings and applies a stack of self-attention layers to produce contextual representations:

$$\mathbf{H} = (\mathbf{h}_1, \ldots, \mathbf{h}_L), \quad \mathbf{h}_i \in \mathbb{R}^d.$$

For sequence classification, the representation of the special classification token (often the first token) is used as a summary:

$$\mathbf{h}_{cls} \in \mathbb{R}^d.$$

A linear classification head produces logits for the two classes:

$$\mathbf{z} = \mathbf{W}\mathbf{h}_{cls} + \mathbf{b}, \quad \mathbf{z} \in \mathbb{R}^2,$$

and probabilities are obtained by softmax:

$$p(y = 1 \mid x') = \frac{e^{z_1}}{e^{z_0} + e^{z_1}}.$$

These probabilities are later converted into binary predictions using a tuned threshold selected to maximize OOF F1.

MacBERT differs from the original BERT primarily in its pretraining masking strategy. Instead of using a mask token that creates an artificial discrepancy between pretraining and fine-tuning, MacBERT adopts a correction-style objective that replaces tokens in a more realistic manner during pretraining. Intuitively, this encourages the encoder to learn representations that better support error correction and natural token recovery. This is helpful for user generated Chinese reviews, which often contain informal expressions, inconsistent punctuation, and noisy phrasing. In our setting, this robustness is valuable because many food safety complaints are expressed indirectly and may not follow standard formal writing patterns.

From an efficiency standpoint, the base sized backbone offers a strong tradeoff between accuracy and runtime on a 12 GB GPU. We further constrain memory and training time by using a moderate maximum sequence length ($L = 192$), FP16 mixed precision training, and gradient accumulation to maintain a stable effective batch size.

3.5 Loss Function with Class Weights

Because the positive class is typically less frequent, we use weighted cross entropy. For each fold, compute class counts n_0, n_1 on the fold training split, total $N = n_0 + n_1$, and weights:

$$w_c = \frac{N}{2 \cdot \max(n_c, 1)}, \quad c \in \{0, 1\}.$$

Then the loss for a single sample is:

$$\mathcal{L} = -w_y \log p(y|x').$$

This increases gradient contribution from the minority class and typically improves recall, which is important under F1.

3.6 Stratified K-fold Cross Validation and OOF Predictions (Mathematical Form)

Let the training set be

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}, \quad y_i \in \{0, 1\}.$$

Stratified K-fold cross validation partitions the index set $\{1, \ldots, N\}$ into K disjoint subsets

$$\{\mathcal{I}_k\}_{k=1}^{K}, \quad \mathcal{I}_k \cap \mathcal{I}_\ell = \emptyset \ (k \neq \ell), \quad \bigcup_{k=1}^{K} \mathcal{I}_k = \{1, \ldots, N\},$$

such that the class proportions are approximately preserved in each fold:

$$\frac{1}{|\mathcal{I}_k|} \sum_{i \in \mathcal{I}_k} \mathbb{I}[y_i = 1] \approx \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}[y_i = 1], \quad \forall k \in \{1, \ldots, K\}.$$

For fold k, define the training indices and validation indices as

$$\mathcal{T}_k = \{1, \ldots, N\} \setminus \mathcal{I}_k, \qquad \mathcal{V}_k = \mathcal{I}_k.$$

Let $f_k(\cdot; \theta_k)$ denote the model trained on fold k using only samples in $\mathcal{T}_k$:

$$\theta_k \in \arg\min_{\theta} \sum_{i \in \mathcal{T}_k} \ell(y_i, f(x_i; \theta)),$$

where $\ell(\cdot, \cdot)$ is the (weighted) cross entropy loss and $f(x_i; \theta)$ outputs class probabilities. Denote the predicted probability for the positive class by

$$p_k(x) = \Pr(y = 1 \mid x; \theta_k).$$

The out-of-fold (OOF) probability vector $p^{oof} \in [0, 1]^N$ is defined component-wise as

$$p_i^{oof} = p_k(x_i), \quad \text{for the unique } k \text{ such that } i \in \mathcal{V}_k.$$

Equivalently,

$$p_i^{oof} = \sum_{k=1}^{K} \mathbb{I}[i \in \mathcal{V}_k] \cdot p_k(x_i).$$

This construction guarantees that for every i, the prediction p_i^{oof} is produced by a model that did not train on sample i, which reduces optimistic bias and supports robust threshold tuning.

Test-time Probability Averaging Across Folds

Let the test set be $\mathcal{D}^{test} = \{x_j^{test}\}_{j=1}^{M}$. Each fold model produces probabilities

$$p_k^{test}(x_j^{test}) = \Pr(y = 1 \mid x_j^{test}; \theta_k).$$

The final test probability is the fold-average:

$$p_j^{test} = \frac{1}{K} \sum_{k=1}^{K} p_k^{test}(x_j^{test}), \quad j = 1, \ldots, M.$$

Threshold Selection on OOF Predictions

Given a threshold $t \in (0, 1)$, define predictions

$$\hat{y}_i(t) = \mathbb{I}[p_i^{oof} \geq t].$$

Let $TP(t), FP(t), FN(t)$ be the resulting counts. Then

$$\text{Precision}(t) = \frac{TP(t)}{TP(t) + FP(t)}, \quad \text{Recall}(t) = \frac{TP(t)}{TP(t) + FN(t)},$$

$$F1(t) = \frac{2\,\text{Precision}(t)\,\text{Recall}(t)}{\text{Precision}(t) + \text{Recall}(t)}.$$

The selected threshold is

$$t^\star \in \arg\max_{t \in \mathcal{G}} F1(t),$$

where $\mathcal{G}$ is a predefined grid of thresholds (for example, 401 evenly spaced values in $[0.01, 0.99]$). Final binary predictions on the test set are

$$\hat{y}_j^{test} = \mathbb{I}[p_j^{test} \geq t^\star], \quad j = 1, \ldots, M.$$

3.7 Cross Validation Strategy

We use stratified K fold cross validation with $K = 5$. Stratification preserves label proportions across folds, which improves stability. For each fold:

- Train on $K - 1$ folds.
- Validate on the held out fold to obtain fold level probabilities.
- Predict on the test set.

OOF probabilities are formed by concatenating the validation predictions from all folds. Test probabilities are averaged across folds.

3.8 Training Configuration and Efficiency

The training configuration is designed for a 12 GB GPU:

- FP16 mixed precision to reduce memory and speed up compute.
- Gradient accumulation to increase effective batch size without exceeding VRAM.
- Early stopping with patience 2 to limit over training and wasted compute.
- Maximum sequence length 192 to balance context and throughput.

These decisions are practical for competition workflows because they reduce runtime while preserving strong model quality.

3.9 Algorithms

Algorithm 1 Stratified Cross Validation with OOF and Test Probability Aggregation

```
Training set $\{(x_i, y_i)\}_{i=1}^{N}$, test set $\{x_j^{test}\}_{j=1}^{M}$, folds $K$
OOF probabilities $p^{oof}$, test probabilities $p^{test}$
Split indices into stratified folds $\{\mathcal{F}_k\}_{k=1}^{K}$
Initialize $p^{oof} \leftarrow \mathbf{0}_N$, list $\mathcal{P}^{test} \leftarrow [\,]$
for $k = 1$ to $K$ do
    $\mathcal{T}_k \leftarrow \{1, \ldots, N\} \setminus \mathcal{F}_k$
    Train model on $\mathcal{T}_k$ with weighted cross entropy
    Predict $p_k^{val}$ on $\mathcal{F}_k$ and set $p^{oof}[\mathcal{F}_k] \leftarrow p_k^{val}$
    Predict $p_k^{test}$ on test and append to $\mathcal{P}^{test}$
end for
$p^{test} \leftarrow \frac{1}{K} \sum_{k=1}^{K} p_k^{test}$
return $p^{oof}, p^{test}$
```

Algorithm 2 Threshold Optimization on OOF Probabilities

```
Labels $y \in \{0,1\}^N$, OOF probabilities $p^{oof} \in [0,1]^N$, grid size $S$
Best threshold $t^\star$
Define threshold grid $\mathcal{T}$ with $S$ values in $[0.01, 0.99]$
$t^\star \leftarrow 0.5$, $F1^\star \leftarrow -\infty$
for each $t \in \mathcal{T}$ do
    $\hat{y}(t) \leftarrow \mathbb{I}\left[p^{oof} \geq t\right]$
    Compute $F1(t)$ on $(y, \hat{y}(t))$
    if $F1(t) > F1^\star$ then
        $F1^\star \leftarrow F1(t)$, $t^\star \leftarrow t$
    end if
end for
return $t^\star$
```

The overall workflow of the proposed architecture is shwon in the Figure 1.

4 Results and Discussion

4.1 OOF Confusion Matrix and Core Metrics

The reported OOF confusion matrix is 2:

$$\begin{bmatrix} TN & FP \\ FN & TP \end{bmatrix} = \begin{bmatrix} 8427 & 62 \\ 172 & 1339 \end{bmatrix}.$$

From these counts:

$$\text{Precision} = \frac{1339}{1339+62} = 0.9557, \quad \text{Recall} = \frac{1339}{1339+172} = 0.8862,$$

$$F1 = \frac{2 \cdot 0.9557 \cdot 0.8862}{0.9557 + 0.8862} = 0.9196,$$

$$\text{Accuracy} = \frac{8427+1339}{8427+62+172+1339} = 0.9766.$$

Table 1: OOF Metrics at the Best Threshold

Metric	Accuracy	Precision	Recall	F1
OOF	0.9766	0.9557	0.8862	0.9196

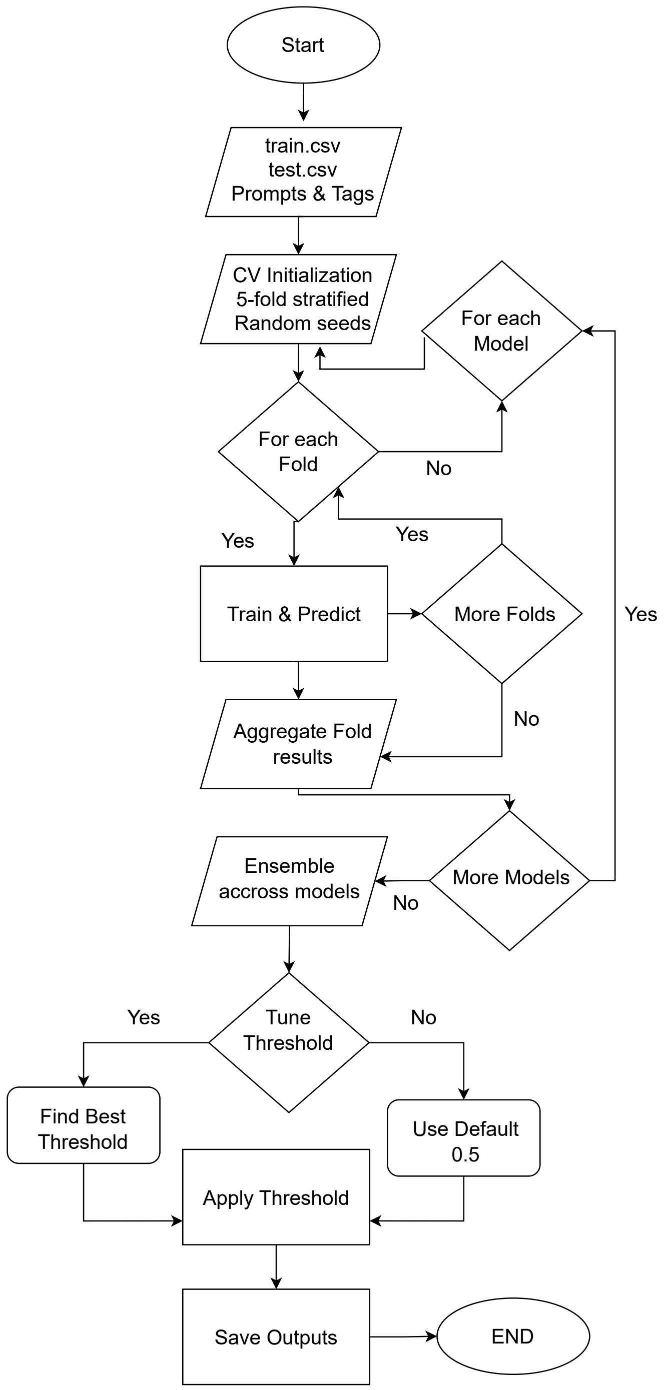


Figure 1: Overall workflow of food review classifier.

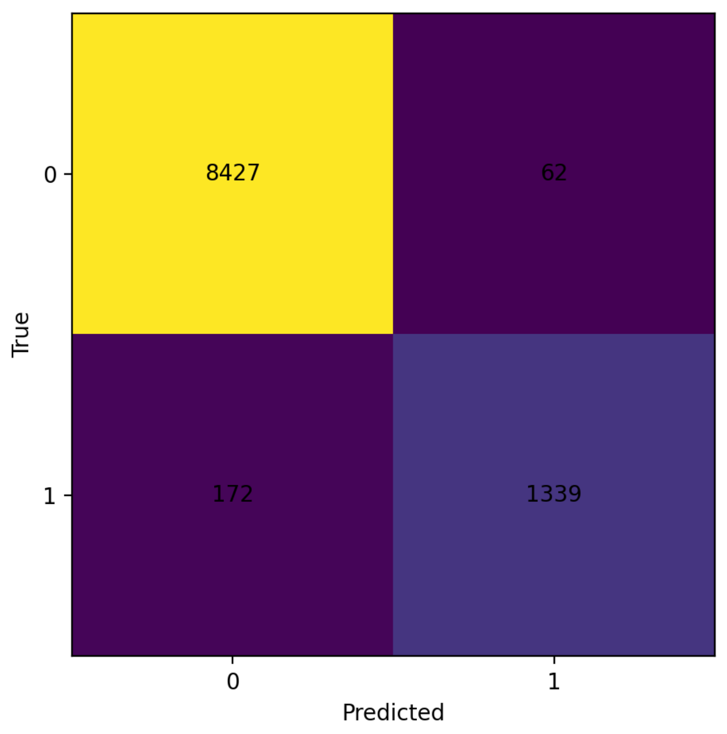


Figure 2: OOF confusion matrix at the selected threshold.

4.2 F1 versus Threshold

The F1 versus threshold curve is a direct visualization of the metric tradeoff 3. When the threshold is low, the classifier predicts more positives, which typically increases recall but can reduce precision due to more false positives. When the threshold is high, the classifier becomes conservative, typically increasing precision but reducing recall due to more false negatives. The OOF search chooses a threshold that balances these effects under the F1 objective. This is why tuning the threshold is often a high return improvement in F1 based competitions.

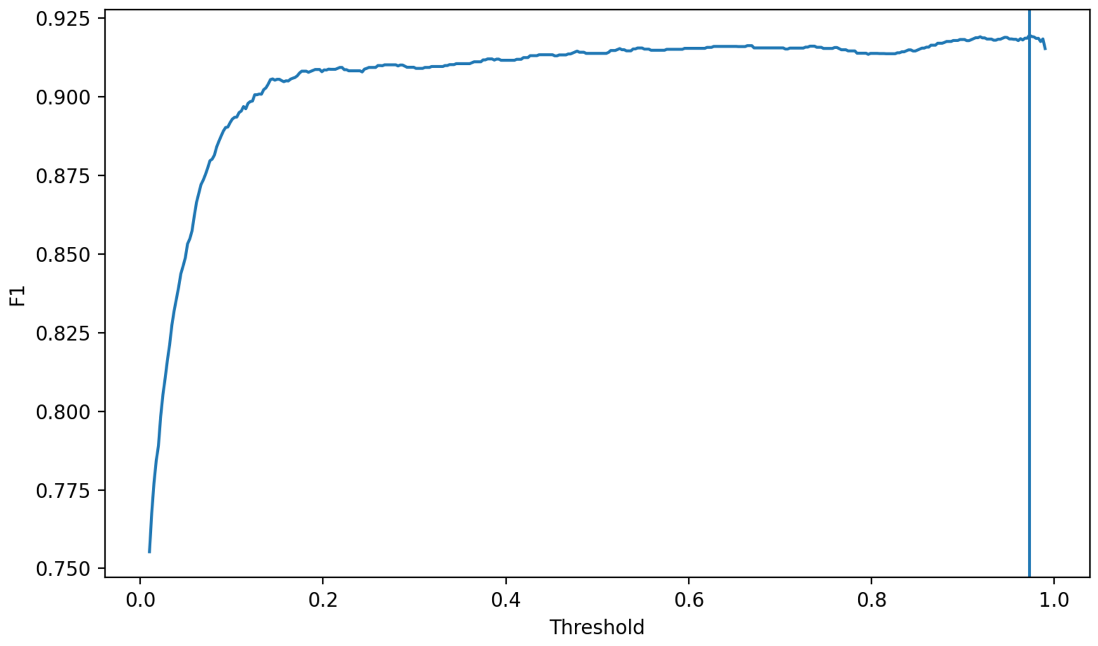


Figure 3: OOF F1 versus threshold.

4.3 ROC Curve and Discrimination

The ROC curve reports an AUC of 0.9911 4. This indicates that the model ranks positive examples above negative examples with very high probability. In practical terms, this suggests the representation learned by the transformer is strong and that errors are more related to boundary cases than systematic confusion.

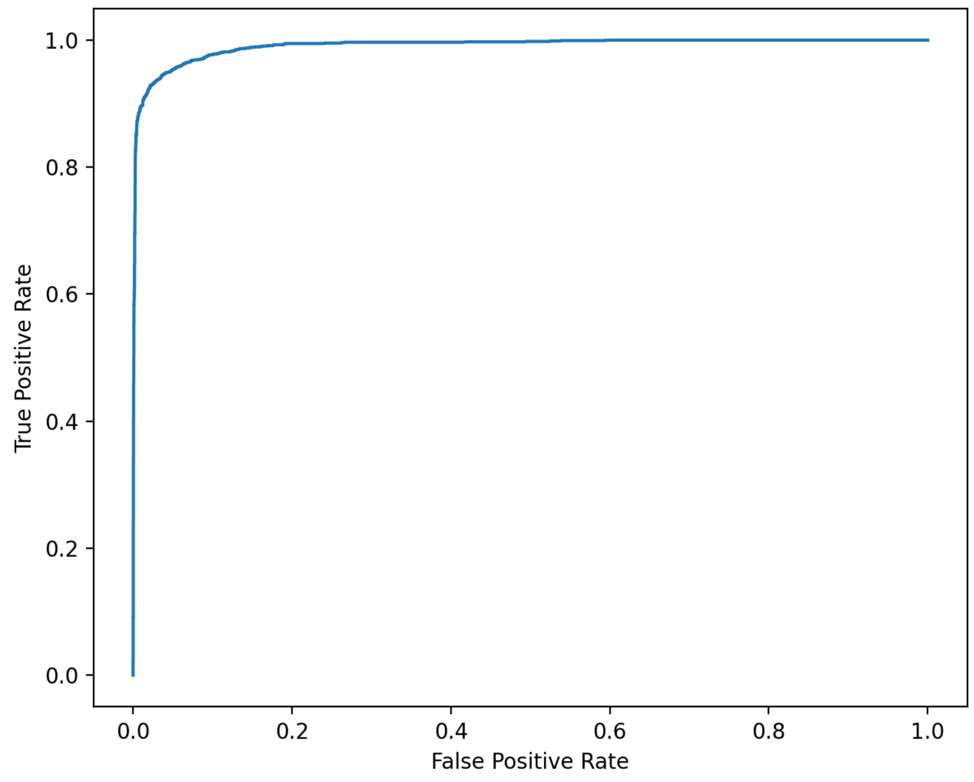


Figure 4: ROC curve with AUC 0.9911.

4.4 Precision Recall Curve

The Precision Recall curve reports an average precision of 0.9633 5. This is a more informative measure than ROC AUC under class imbalance. The curve shape suggests that high precision is maintained across a wide range of recall values, which is consistent with the low number of false positives observed in the confusion matrix.

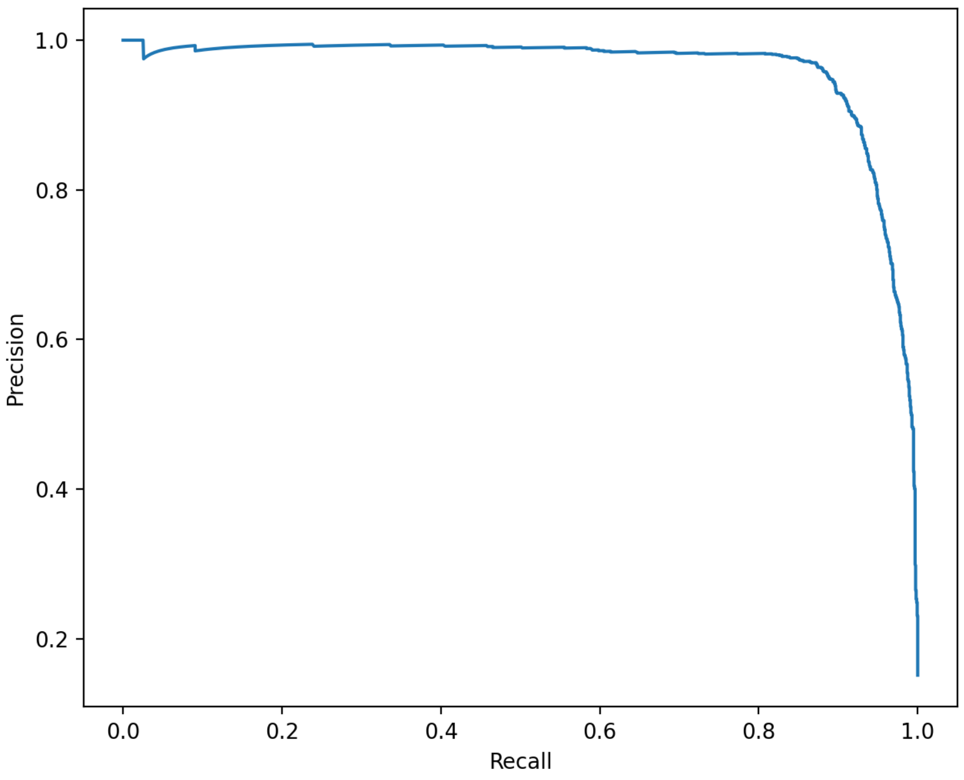


Figure 5: Precision Recall curve with AP 0.9633.

4.5 Calibration Curve and Threshold Stability

The calibration curve shows how predicted probabilities map to empirical positive rates 6. Deviations from the diagonal indicate miscalibration. Even with strong discrimination, miscalibration can cause threshold instability when the test distribution differs from the OOF distribution. This explains why leaderboard improvements often come from ensembling and calibration, even when OOF metrics are already high.

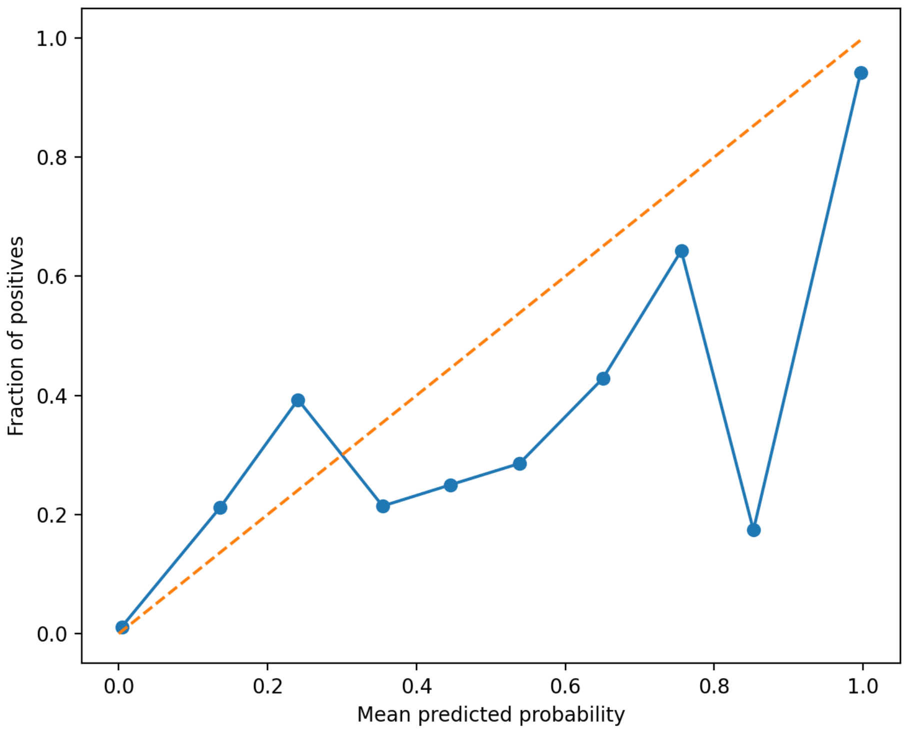


Figure 6: Calibration curve for OOF probabilities.

4.6 Training and Validation Behavior

The training curves show decreasing training loss with a rising evaluation loss in later epochs 7. This can happen when the model becomes more confident. Cross entropy punishes confident mistakes heavily. If a small number of hard or noisy validation examples remain incorrect while confidence increases, the validation loss can rise even when classification metrics remain stable. The evaluation F1 curve is therefore more aligned with the target objective than evaluation loss in this setting.

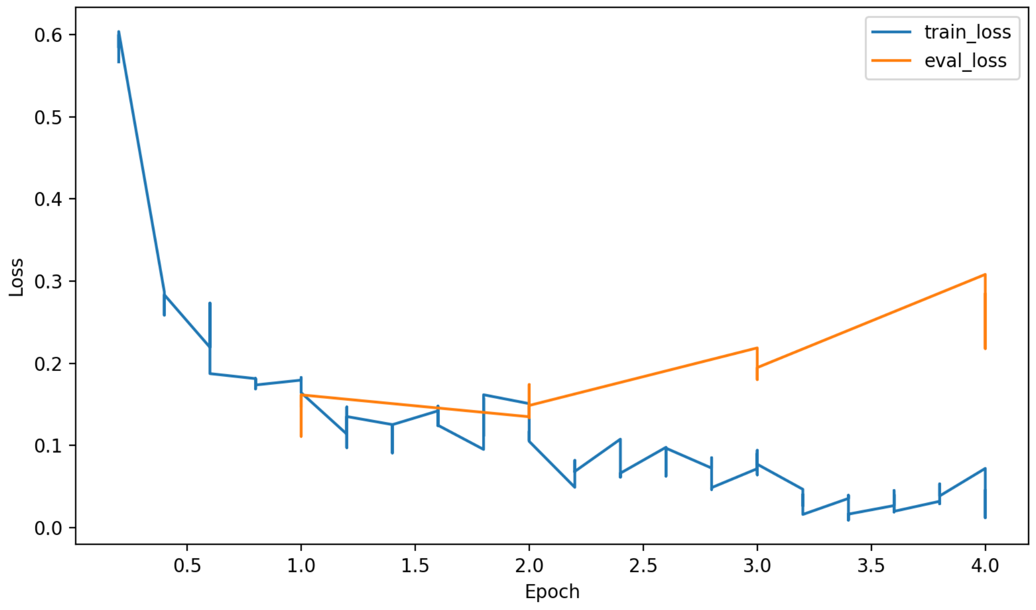


Figure 7: Aggregated training and evaluation loss across folds.

4.7 Per Fold Stability

Per fold F1 at threshold 0.5 is consistent across folds, which suggests the system is not overly dependent on a single split 8. This stability is important for competition generalization because it reduces the chance that performance is inflated by a fortunate validation partition.

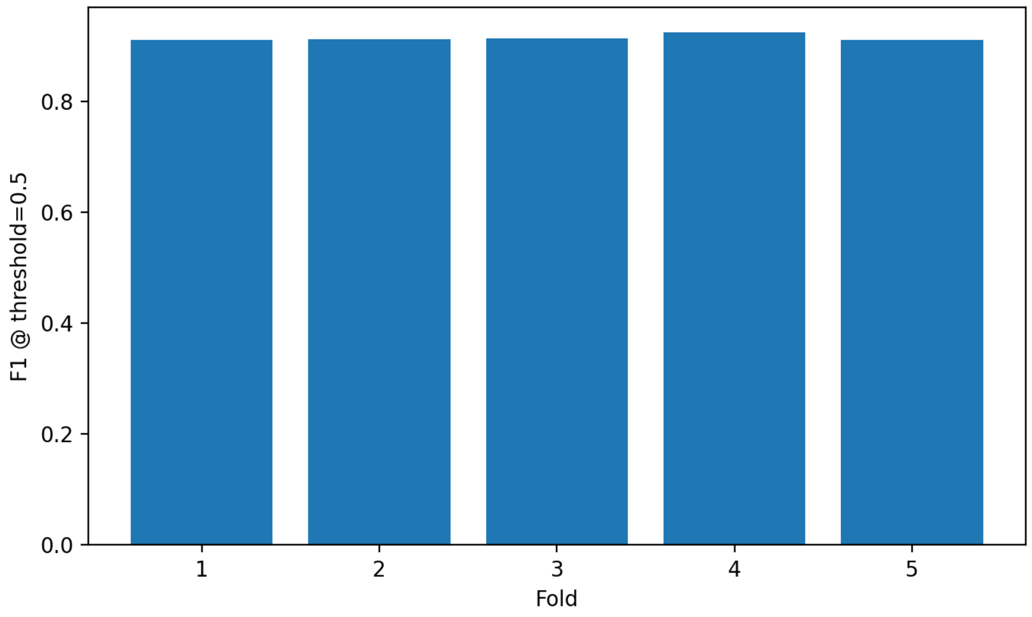


Figure 8: Per fold F1 at threshold 0.5.

4.8 Where Errors Come From

Based on the confusion matrix:

- **False positives (62)**: These are likely reviews that mention keywords or negative sentiment but are not genuine safety complaints. Examples might include complaints about taste or delivery being misinterpreted as hygiene risk.
- **False negatives (172)**: These are likely subtle complaints that do not contain obvious keywords, or use indirect language. These cases often benefit from additional context, richer augmentation, or a slightly more recall oriented threshold.

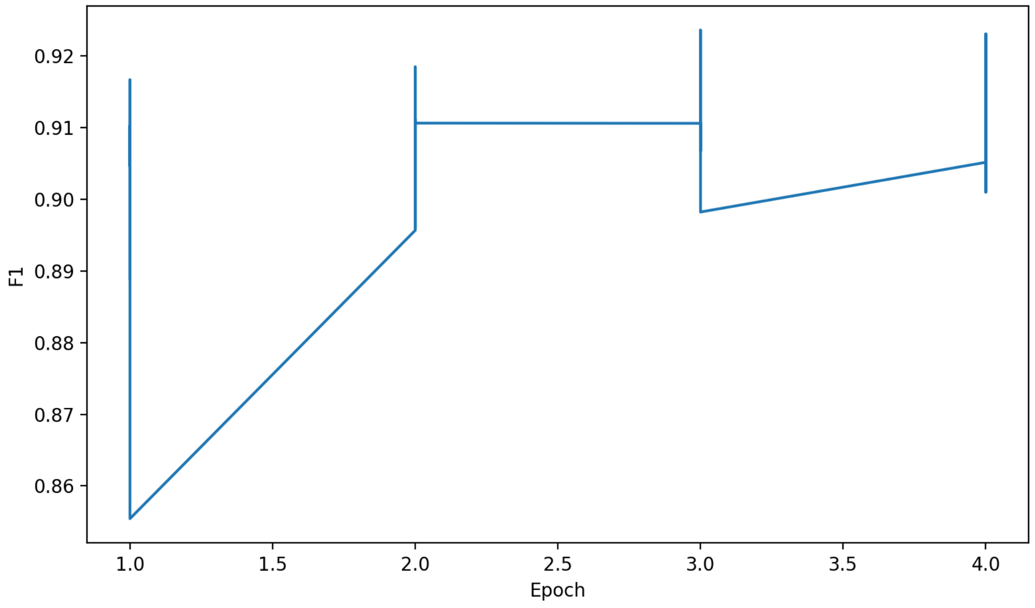
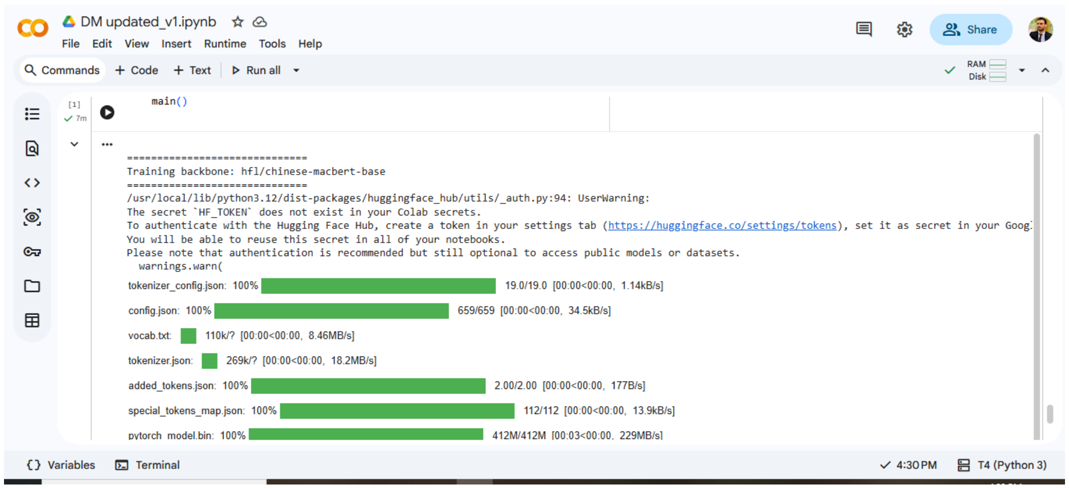


Figure 9: Aggregated evaluation F1 across folds by epoch.

4.9 Training and Rank

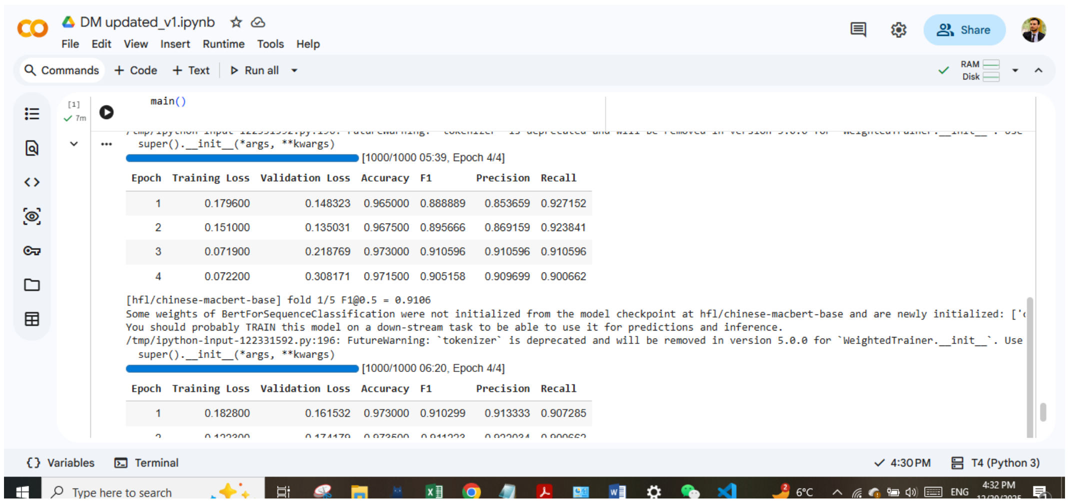
Some of the screenshots of the training and experiment are given below, along with the rank during the competition.



```
main()
```

```
==============================
Training backbone: hfl/chinese-macbert-base
==============================
/usr/local/lib/python3.12/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Goog
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
tokenizer_config.json: 100%  19.0/19.0 [00:00<00:00, 1.14kB/s]
config.json: 100%  659/659 [00:00<00:00, 34.5kB/s]
vocab.txt:  110k/? [00:00<00:00, 8.46MB/s]
tokenizer.json:  269k/? [00:00<00:00, 18.2MB/s]
added_tokens.json: 100%  2.00/2.00 [00:00<00:00, 177B/s]
special_tokens_map.json: 100%  112/112 [00:00<00:00, 13.9kB/s]
pytorch_model.bin: 100%  412M/412M [00:03<00:00, 229MB/s]
```

Figure 10: Model loading process.



```
main()
```

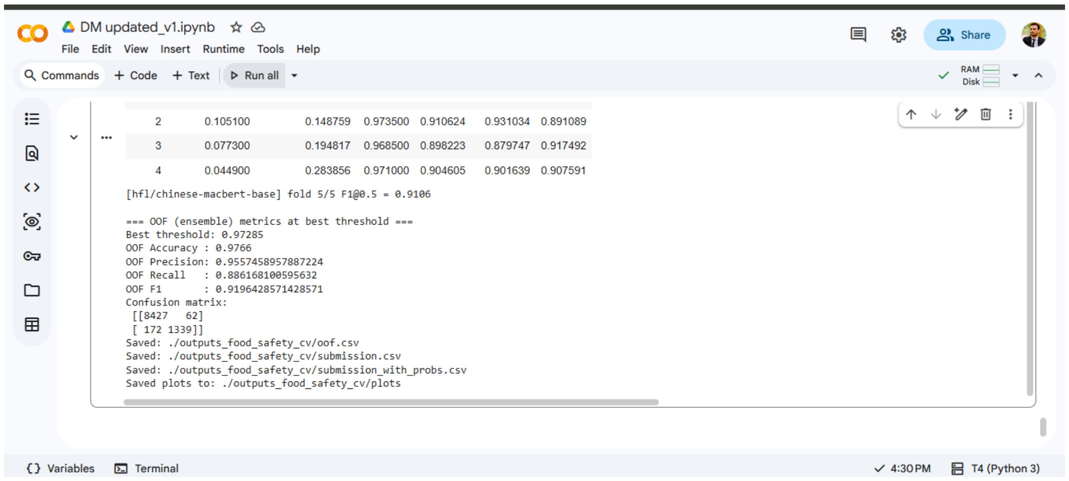
```
  super().__init__(*args, **kwargs)
[1000/1000 05:39, Epoch 4/4]
```

Epoch	Training Loss	Validation Loss	Accuracy	F1	Precision	Recall
1	0.179600	0.148323	0.965000	0.888889	0.853659	0.927152
2	0.151000	0.135031	0.967500	0.895666	0.869159	0.923841
3	0.071900	0.218769	0.973000	0.910596	0.910596	0.910596
4	0.072200	0.308171	0.971500	0.905158	0.909699	0.900662

```
[hfl/chinese-macbert-base] fold 1/5 F1@0.5 = 0.9106
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at hfl/chinese-macbert-base and are newly initialized: ['
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
/tmp/ipython-input-122331592.py:196: FutureWarning: `tokenizer` is deprecated and will be removed in version 5.0.0 for `WeightedTrainer.__init__`. Use
  super().__init__(*args, **kwargs)
[1000/1000 06:20, Epoch 4/4]
```

Epoch	Training Loss	Validation Loss	Accuracy	F1	Precision	Recall
1	0.182800	0.161532	0.973000	0.910299	0.913333	0.907285

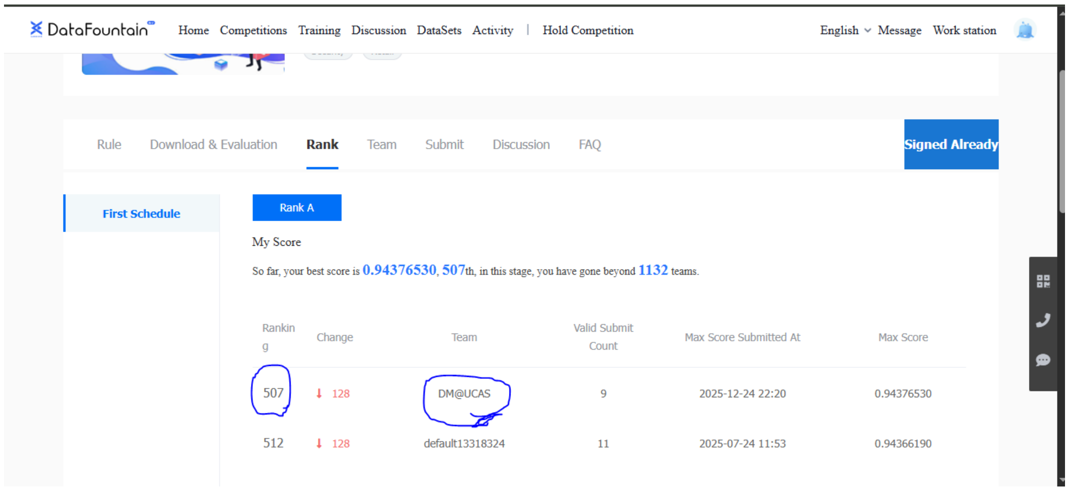
Figure 11: Training process.



```
   2    0.105100    0.148759  0.973500  0.910624   0.931034  0.891089
   3    0.077300    0.194817  0.968500  0.898223   0.879747  0.917492
   4    0.044900    0.283856  0.971000  0.904605   0.901639  0.907591
[hfl/chinese-macbert-base] fold 5/5 F1@0.5 = 0.9106

=== OOF (ensemble) metrics at best threshold ===
Best threshold: 0.97285
OOF Accuracy : 0.9766
OOF Precision: 0.9557458957887224
OOF Recall   : 0.886168100595632
OOF F1       : 0.9196428571428571
Confusion matrix:
 [[8427   62]
 [ 172 1339]]
Saved: ./outputs_food_safety_cv/oof.csv
Saved: ./outputs_food_safety_cv/submission.csv
Saved: ./outputs_food_safety_cv/submission_with_probs.csv
Saved plots to: ./outputs_food_safety_cv/plots
```

Figure 12: Final output.


DataFountain
Home Competitions Training Discussion DataSets Activity | Hold Competition
English Message Work station
Rule Download & Evaluation Rank Team Submit Discussion FAQ
Signed Already
First Schedule
Rank A

My Score

So far, your best score is 0.94376530, 507th, in this stage, you have gone beyond 1132 teams.

Ranking	Change	Team	Valid Submit Count	Max Score Submitted At	Max Score
507	↓ 128	DM@UCAS	9	2025-12-24 22:20	0.94376530
512	↓ 128	default13318324	11	2025-07-24 11:53	0.94366190

Figure 13: Final rank in the competition.

5 Conclusion

This report presented a practical and competition aligned approach for Chinese food safety complaint detection from reviews. The system combines a strong transformer backbone with stratified cross validation, weighted loss for class imbalance, and OOF threshold tuning for the positive class F1 score. Diagnostics show excellent separability, with ROC AUC of 0.9911 and average precision of 0.9633, and strong OOF classification performance with an F1 of 0.9196 and accuracy of 0.9766. The confusion matrix indicates low false positive rates and moderate false negatives, which is typical for safety related detection where indirect phrasing is common. The plots provide transparency into threshold tradeoffs, discrimination, calibration, and training dynamics. Overall, the approach is efficient on a 12 GB GPU and forms a solid foundation for further leaderboard improvements through stronger ensembling, calibration methods, and targeted strategies to reduce false negatives without increasing false positives too much.

5.1 Future Work

We use the base-size backbone `hfl/chinese-macbert-base` to match the available compute budget. MacBERT also has larger variants (for example, `MacBERT-large`) with increased depth and parameter count. With sufficient GPU memory and training time, these higher-capacity backbones can potentially yield improved F1 by learning richer representations, at the cost of slower fine-tuning and higher inference latency. Therefore, our current choice prioritizes an efficient accuracy–compute balance, while larger variants remain a straightforward upgrade path under a stronger compute setting.