




# LabelFusion: Learning to Fuse LLMs and Transformer Classifiers for Robust Text Classification

Christoph Weisser<sup>1,2</sup> and LabelFusion contributors<sup>1</sup>

<sup>1</sup> Campus-Institut Data Science, Göttingen, Germany <sup>2</sup> Centre for Statistics, Georg-August-Universität Göttingen, Germany

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Open Journals](#) 

## Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

## Summary

LabelFusion is a fusion ensemble for text classification that learns to combine a traditional transformer-based classifier (e.g., RoBERTa) with one or more Large Language Models (LLMs) such as OpenAI GPT, Google Gemini, or DeepSeek) to deliver accurate and cost-aware predictions across multi-class and multi-label tasks. The package provides a simple high-level interface (AutoFusionClassifier) that trains the full pipeline end-to-end with minimal configuration, and a flexible API for advanced users. Under the hood, LabelFusion concatenates vector signals from the ML backbone (logits) and LLM(s) (per-class scores) and trains a compact multi-layer perceptron (FusionMLP) to produce the final prediction. This learned fusion approach captures complementary strengths of LLM reasoning and transformer efficiency, yielding robust performance across domains—achieving 92.4% accuracy on AG News topic classification—while enabling practical trade-offs between accuracy, latency, and cost.

## Statement of Need

Modern text classification spans diverse scenarios—from sentiment and topic tagging to policy enforcement and routing—often under constraints that vary per deployment (throughput, cost ceilings, data privacy). While transformer classifiers such as BERT/RoBERTa achieve strong supervised performance (Devlin et al., 2018; Liu et al., 2019), frontier LLMs can excel in low-data, ambiguous, or cross-domain settings (OpenAI, 2023). No single model family is uniformly best: LLMs are powerful yet comparatively costly and rate-limited, whereas fine-tuned transformers are efficient but may struggle with out-of-distribution cases.

LabelFusion addresses this gap by: (1) exposing a minimal “AutoFusion” interface that trains a learned combination of an ML backbone and one or more LLMs; (2) supporting both multi-class and multi-label classification; (3) providing a lightweight fusion learner that directly fits on LLM scores and ML logits; and (4) integrating cleanly with existing ensemble utilities. Researchers and practitioners can therefore leverage LLMs where they add value while retaining the speed and determinism of transformer models.

## State of the Field

Ensembles improve robustness by aggregating diverse predictors (Dietterich, 2000; Hansen & Salamon, 1990). Mixture-of-experts approaches further specialize components and learn to combine their outputs (Jacobs et al., 1991). In applied NLP, common tools such as scikit-learn (Pedregosa et al., 2011) and Hugging Face Transformers (Wolf et al., 2019) offer strong baselines but do not provide a turnkey, learned fusion of LLMs with supervised transformers. Orchestration frameworks (e.g., LangChain) focus on tool use rather than classification ensembles. LabelFusion contributes a focused, production-minded implementation of a small learned combiner that operates on per-class signals from both model families.

## 41 Functionality and Design

42 LabelFusion consists of three layers:

- 43     ▪ ML component: a RoBERTa-style classifier produces per-class logits for input texts.
- 44     ▪ LLM component(s): provider-specific classifiers (OpenAI, Gemini, DeepSeek) return
- 45       per-class scores via prompting. Scores can be cached to minimize API calls when cache
- 46       locations are provided.
- 47     ▪ Fusion component: a compact MLP concatenates ML logits and LLM scores and outputs
- 48       fused logits. The ML backbone is trained/fine-tuned with a small learning rate; the fusion
- 49       MLP uses a higher rate, enabling rapid adaptation without destabilizing the encoder.

50 Key features:

- 51     ▪ **Multi-class and multi-label support** with consistent data structures and unified training
- 52       pipeline.
- 53     ▪ **Optional LLM response caching** reuses on-disk predictions when cache paths are supplied,
- 54       with dataset-hash validation to guard against stale files.
- 55     ▪ **Batched scoring** processes multiple texts efficiently with configurable batch sizes for both
- 56       ML tokenization and LLM API calls.
- 57     ▪ **Results management** via ResultsManager tracks experiments, stores predictions, com-
- 58       putes metrics, and enables reproducible research workflows.
- 59     ▪ **Flexible interfaces:** Command-line training via `train_fusion.py` with YAML configs for
- 60       research; or minimal AutoFusion API for quick deployment.
- 61     ▪ **Composable design:** LabelFusion can serve as a strong base learner in higher-level
- 62       ensembles (e.g., voting/weighted combinations of multiple fusion models).

## 63 Minimal Example (AutoFusion)

```
from textclassify import AutoFusionClassifier
```

```
config = {  
    'llm_provider': 'deepseek',  
    'label_columns': ['positive', 'negative', 'neutral']  
}
```

```
clf = AutoFusionClassifier(config)
```

```
clf.fit(train_dataframe)
```

```
pred = clf.predict(["This is amazing!"]) # fused prediction
```

## 64 CLI and Configuration

65 Users can generate a starter config and train via the command line:

- 66     ▪ Create config: `python train_fusion.py --create-config fusion_config.yaml`
- 67     ▪ Train: `python train_fusion.py --config fusion_config.yaml`
- 68     ▪ Optional test data and output artifacts are also supported.

## 69 Quality Control

70 The repository ships legacy unit tests under `tests/evaluation/old/` that cover configuration

71 handling, core types, and package integration. Fusion-specific logic is currently exercised

72 through CLI-driven workflows and notebooks that run end-to-end training with deterministic

73 seeds where applicable.

74 Evaluation scripts (`tests/evaluation/`) provide comprehensive benchmarking on standard

75 datasets: - **AG News** (Zhang et al., 2015): 4-class topic classification with experiments

76 across varying training data sizes (20%–100%) - **GoEmotions** (Demszky et al., 2020): 28-class

77 multi-label emotion classification for validating multi-label fusion performance

78 LLM scoring paths implement retries and disk caching; transformer training supports standard  
79 sanity checks (overfit a small batch, reduced batch sizes for constrained hardware). Metrics  
80 (accuracy/F1, per-label scores) are computed automatically and stored with run artifacts to  
81 facilitate regression tracking and reproducibility.

## 82 Availability and Installation

83 LabelFusion is distributed as part of the textclassify package under the MIT license and  
84 is available at <https://github.com/DataandAIResearch/LabelFusion>. The fusion components  
85 require Python 3.8+ and common scientific Python dependencies (PyTorch, transformers,  
86 scikit-learn, numpy, pandas, PyYAML). Optional plotting depends on matplotlib/seaborn.  
87 Installation and quick-start snippets are provided in the README and FUSION\_README.md.

## 88 Production-Ready Features

89 Beyond the core fusion methodology, LabelFusion includes features for practical deployment:

- 90 ■ **LLM Response Caching:** Optional disk-backed caches reuse prior predictions when cache  
91 paths are supplied, with dataset hashes to flag inconsistent inputs.
- 92 ■ **Results Management:** Built-in ResultsManager tracks experiments, stores predictions,  
93 and computes metrics automatically. Supports comparison across runs and configuration  
94 tracking.
- 95 ■ **Batch Processing:** Efficient batched scoring of texts with configurable batch sizes for  
96 both ML and LLM components.

## 97 Impact and Use Cases

### 98 Empirical Performance

99 LabelFusion has been evaluated on standard benchmark datasets to validate its effectiveness.  
100 Key findings demonstrate consistent improvements over individual model components:

#### 101 AG News Topic Classification

102 Evaluation on the AG News dataset (Zhang et al., 2015) (4-class topic classification) with  
103 5,000 test samples shows:

Training Data	Model	Accuracy	F1-Score	Precision	Recall
20% (800)	<b>Fusion</b>	<b>92.2%</b>	<b>0.922</b>	0.923	0.922
20% (800)	RoBERTa	89.8%	0.899	0.902	0.898
20% (800)	OpenAI	84.4%	0.844	0.857	0.844
40% (1,600)	<b>Fusion</b>	<b>92.2%</b>	<b>0.922</b>	0.924	0.922
40% (1,600)	RoBERTa	91.0%	0.911	0.913	0.910
40% (1,600)	OpenAI	84.4%	0.844	0.857	0.844
100% (4,000)	<b>Fusion</b>	<b>92.4%</b>	<b>0.924</b>	0.926	0.924
100% (4,000)	RoBERTa	92.2%	0.922	0.923	0.922
100% (4,000)	OpenAI	84.4%	0.844	0.857	0.844

104 **Key Observations:** - Fusion consistently outperforms individual models across all training  
105 data sizes - With only 20% training data, Fusion achieves 92.2% accuracy—matching its  
106 performance with full data - Demonstrates superior **data efficiency**: fusion learning extracts  
107 maximum value from limited examples - RoBERTa alone requires 100% of data to approach  
108 Fusion's 20% performance - LLM (OpenAI) shows stable but lower performance, highlighting  
109 the value of combining approaches

110 These results validate that learned fusion captures complementary strengths: the LLM provides  
111 robust reasoning even with limited training data, while the ML backbone adds efficiency and  
112 domain-specific patterns.

### 113 Application Domains

114 Learned fusion excels in scenarios where model strengths complement each other:

- 115 ■ **Customer feedback analysis** with nuanced multi-label taxonomies where LLMs handle  
116 ambiguous sentiment while ML models efficiently process clear cases
- 117 ■ **Content moderation** where uncertain cases benefit from LLM reasoning while rou-  
118 tine items rely on the fast ML backbone, enabling real-time processing with accuracy  
119 guarantees
- 120 ■ **Scientific literature classification** across heterogeneous topics where domain shift is  
121 common and LLMs provide robustness to new terminology
- 122 ■ **Low-resource settings** where limited training data is available but task complexity requires  
123 sophisticated reasoning

124 The approach enables pragmatic cost control (e.g., the fusion layer learns when to rely more  
125 heavily on the efficient ML backbone versus the more expensive LLM signal) while retaining a  
126 single trainable decision surface that optimizes for the specific deployment constraints.

### 127 Acknowledgements

128 We thank contributors and users who reported issues and shared datasets. LabelFusion  
129 builds on the open-source ecosystem, notably Hugging Face Transformers (Wolf et al., 2019),  
130 scikit-learn (Pedregosa et al., 2011), PyTorch (Paszke et al., 2019), and LLM provider SDKs.  
131 We acknowledge the use of the AG News and GoEmotions benchmark datasets for evaluation.

### 132 References

- 133 Demszky, D., Movshovitz-Attias, D., Ko, J., Cowen, A., Nemade, G., & Ravi, S. (2020).  
134 GoEmotions: A dataset of fine-grained emotions. *Proceedings of the 58th Annual Meeting*  
135 *of the Association for Computational Linguistics*, 4040–4054.
- 136 Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep  
137 bidirectional transformers for language understanding. *arXiv Preprint arXiv:1810.04805*.
- 138 Dietterich, T. G. (2000). Ensemble methods in machine learning. *International Workshop on*  
139 *Multiple Classifier Systems*, 1–15.
- 140 Hansen, L. K., & Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on*  
141 *Pattern Analysis and Machine Intelligence*, 12(10), 993–1001.
- 142 Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of  
143 local experts. *Neural Computation*, 3(1), 79–87.
- 144 Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer,  
145 L., & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach.  
146 *arXiv Preprint arXiv:1907.11692*.
- 147 OpenAI. (2023). GPT-4 technical report. *arXiv Preprint arXiv:2303.08774*.
- 148 Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin,  
149 Z., Gimelshein, N., Antiga, L., & others. (2019). PyTorch: An imperative style, high-  
150 performance deep learning library. *Advances in Neural Information Processing Systems*,  
151 32.
- 152 Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M.,  
153 Prettenhofer, P., Weiss, R., Dubourg, V., & others. (2011). Scikit-learn: Machine learning

- 154        in python. *Journal of Machine Learning Research*, 12(Oct), 2825–2830.
- 155        Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T.,  
156        Louf, R., Funtowicz, M., & others. (2019). HuggingFace's transformers: State-of-the-art  
157        natural language processing. *arXiv Preprint arXiv:1910.03771*.
- 158        Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text  
159        classification. *Advances in Neural Information Processing Systems*, 28, 649–657.

DRAFT