

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

Michael Schlee<sup>1</sup>, Christoph Weisser<sup>1</sup>, Timo Kivimäki<sup>2</sup>, Melchizedek Mashiku<sup>4</sup>, and Benjamin Saefken<sup>3</sup>

<sup>1</sup> Centre for Statistics, Georg-August-Universität Göttingen, Germany <sup>2</sup> Department of Politics and International Studies, University of Bath, Bath, UK <sup>3</sup> Institute of Mathematics, Clausthal University of Technology, Clausthal-Zellerfeld, Germany <sup>4</sup> Tanaq Management Services LLC, Contracting Agency to the Division of Viral Diseases, Centers for Disease Control and Prevention, Chamblee, Georgia, USA

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Published: unpublished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 integrates vector signals from both sources by concatenating the ML backbone's embeddings with the LLM-derived per-class scores—obtained through structured prompt-engineering strategies—and feeds this joint representation into a compact multi-layer perceptron (FusionMLP) that produces the final prediction. This learned fusion approach captures complementary strengths of LLM reasoning and traditional transformer-based classifiers, yielding robust performance across domains—achieving 92.4% accuracy on AG News and 92.3% on 10-class Reuters 21578 topic classification — while enabling practical trade-offs between accuracy, latency, and cost.

## Statement of Need

Modern text classification spans diverse scenarios, from sentiment analysis (Kant et al., 2024; Luber et al., 2021; Thormann et al., 2021) to complex topic tagging (Kant et al., 2022; A. Thielmann, Weisser, Krenz, & Säfken, 2021; A. Thielmann, Weisser, & Krenz, 2021; A. F. Thielmann et al., 2024), 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 typically uniformly best: LLMs are powerful, but comparatively costly, whereas fine-tuned transformers are efficient but may struggle with out-of-distribution cases or extremely limited training examples.

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 embeddings; 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

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 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.

## Functionality and Design

LabelFusion consists of three layers:

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

Key features:

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

We support both multi-class setups (one label per input) and multi-label scenarios (multiple labels per input), and point readers to Appendix A for formal definitions and training implications.

## Minimal Example (AutoFusion)

```
from textclassify.ensemble.auto_fusion import AutoFusionClassifier

# Multi-class: exactly one of the sentiment labels applies
multiclass_config = {
    'llm_provider': 'deepseek',
    'label_columns': ['positive', 'negative', 'neutral'],
    'multi_label': False
}
multiclass_clf = AutoFusionClassifier(multiclass_config)
multiclass_clf.fit(train_dataframe)
multiclass_pred = multiclass_clf.predict(["This is amazing!"])

# Multi-label: news article can belong to several topics simultaneously
multilabel_config = {
    'llm_provider': 'deepseek',
```

```
'label_columns': ['politics', 'economy', 'technology'],  
'multi_label': True  
}  
multilabel_clf = AutoFusionClassifier(multilabel_config)  
multilabel_clf.fit(train_dataframe)  
multilabel_pred = multilabel_clf.predict(["New investment in AI chips"])
```

## Quality Control

The repository ships legacy unit tests under `tests/evaluation/old/` that cover configuration handling, core types, and package integration. Fusion-specific logic is currently exercised through CLI-driven workflows and notebooks that run end-to-end training with deterministic seeds where applicable.

Evaluation scripts (`tests/evaluation/`) provide comprehensive benchmarking on standard datasets: - **AG News** (Zhang et al., 2015): 4-class topic classification with experiments across varying training data sizes (20%–100%) - **Reuters-21578** (Lewis, 1997): A single-label 10-class subset of the Reuters-21578 corpus, used to evaluate multi-class fusion performance on moderately imbalanced news topics.

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

## Availability and Installation

LabelFusion is distributed as part of the `textclassify` package under the MIT license and is available at <https://github.com/DataandAIResearch/LabelFusion>. The fusion components require Python 3.8+ and common scientific Python dependencies (PyTorch, transformers, scikit-learn, numpy, pandas, PyYAML, matplotlib, seaborn). Installation and quick-start snippets are provided in the README.

## Production-Ready Features

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

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

## Impact and Use Cases

### Empirical Performance

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

#### AG News Topic Classification

Evaluation on the AG News dataset (Zhang et al., 2015) (4-class topic classification) with 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        | 85.1%        | 0.847        | 0.863     | 0.846  |
| 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        | 83.9%        | 0.835        | 0.847     | 0.834  |
| 60% (2,400)   | <b>Fusion</b> | <b>92.0%</b> | <b>0.920</b> | 0.922     | 0.920  |
| 60% (2,400)   | RoBERTa       | 91.0%        | 0.910        | 0.911     | 0.910  |
| 60% (2,400)   | OpenAI        | 85.2%        | 0.847        | 0.861     | 0.844  |
| 80% (3,200)   | <b>Fusion</b> | <b>91.6%</b> | <b>0.916</b> | 0.917     | 0.916  |
| 80% (3,200)   | RoBERTa       | 91.4%        | 0.914        | 0.915     | 0.914  |
| 80% (3,200)   | OpenAI        | 84.1%        | 0.837        | 0.849     | 0.832  |
| 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        | 85.3%        | 0.849        | 0.868     | 0.847  |

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

Reuters-21578 Topic Classification

| Training Data | Model         | Accuracy | F1-Score | Precision | Recall |
|---------------|---------------|----------|----------|-----------|--------|
| 20% (1168)    | <b>Fusion</b> | 72.0%    | 0.752    | 0.769     | 0.745  |
| 20% (1168)    | RoBERTa       | 67.3%    | 0.534    | 0.465     | 0.643  |
| 20% (1168)    | OpenAI        | 88.6%    | 0.928    | 0.951     | 0.923  |
| 40% (2336)    | <b>Fusion</b> | 83.6%    | 0.886    | 0.893     | 0.889  |
| 40% (2336)    | RoBERTa       | 82.0%    | 0.836    | 0.858     | 0.850  |
| 40% (2336)    | OpenAI        | 87.9%    | 0.931    | 0.952     | 0.917  |
| 60% (3505)    | <b>Fusion</b> | 85.5%    | 0.932    | 0.929     | 0.950  |
| 60% (3505)    | RoBERTa       | 83.4%    | 0.907    | 0.906     | 0.945  |
| 60% (3505)    | OpenAI        | 88.4%    | 0.938    | 0.959     | 0.924  |
| 80% (4673)    | <b>Fusion</b> | 90.2%    | 0.954    | 0.954     | 0.965  |
| 80% (4673)    | RoBERTa       | 88.8%    | 0.943    | 0.930     | 0.966  |
| 80% (4673)    | OpenAI        | 88.0%    | 0.934    | 0.951     | 0.918  |
| 100% (5842)   | <b>Fusion</b> | 92.3%    | 0.960    | 0.967     | 0.961  |
| 100% (5842)   | RoBERTa       | 89.0%    | 0.946    | 0.932     | 0.966  |
| 100% (5842)   | OpenAI        | 88.9%    | 0.939    | 0.963     | 0.927  |

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

| Training Data | Model         | Accuracy     | F1-Score     | Precision | Recall |
|---------------|---------------|--------------|--------------|-----------|--------|
| 5% (292)      | <b>Fusion</b> | <b>70.6%</b> | <b>0.717</b> | 0.720     | 0.715  |
| 5% (292)      | RoBERTa       | 0.0%         | 0.372        | 0.276     | 0.713  |
| 5% (292)      | OpenAI        | 88.1%        | 0.930        | 0.952     | 0.917  |
| 10% (584)     | <b>Fusion</b> | <b>67.0%</b> | <b>0.671</b> | 0.672     | 0.671  |
| 10% (584)     | RoBERTa       | 40.0%        | 0.417        | 0.321     | 0.616  |
| 10% (584)     | OpenAI        | 88.5%        | 0.938        | 0.962     | 0.926  |
| 20% (1168)    | <b>Fusion</b> | <b>72.0%</b> | <b>0.752</b> | 0.769     | 0.745  |
| 20% (1168)    | RoBERTa       | 67.3%        | 0.534        | 0.465     | 0.643  |
| 20% (1168)    | OpenAI        | 88.6%        | 0.928        | 0.951     | 0.923  |
| 40% (2336)    | <b>Fusion</b> | <b>83.6%</b> | <b>0.886</b> | 0.893     | 0.889  |
| 40% (2336)    | RoBERTa       | 82.0%        | 0.836        | 0.858     | 0.850  |
| 40% (2336)    | OpenAI        | 87.9%        | 0.931        | 0.952     | 0.917  |
| 60% (3505)    | <b>Fusion</b> | <b>85.5%</b> | <b>0.932</b> | 0.929     | 0.950  |
| 60% (3505)    | RoBERTa       | 83.4%        | 0.907        | 0.906     | 0.945  |
| 60% (3505)    | OpenAI        | 88.4%        | 0.938        | 0.959     | 0.924  |
| 80% (4673)    | <b>Fusion</b> | <b>90.2%</b> | <b>0.954</b> | 0.954     | 0.965  |
| 80% (4673)    | RoBERTa       | 88.8%        | 0.943        | 0.930     | 0.966  |
| 80% (4673)    | OpenAI        | 88.0%        | 0.934        | 0.951     | 0.918  |
| 100% (5842)   | <b>Fusion</b> | <b>92.3%</b> | <b>0.960</b> | 0.967     | 0.961  |
| 100% (5842)   | RoBERTa       | 89.0%        | 0.946        | 0.932     | 0.966  |
| 100% (5842)   | OpenAI        | 88.9%        | 0.939        | 0.963     | 0.927  |

**Key Observations:** - In extremely low-data settings, the Fusion Ensembles appear negatively affected by the RoBERTa component, resulting in reduced overall prediction performance - The LLM (OpenAI) is the preferred model in low-data regimes for multi-label classification on the 10-class Reuters-21578 subset - RoBERTa alone requires around 80% of the training data to reach the LLM's performance at only 5% - In high-data settings (80% to 100%), Fusion Ensembles outperform the individual models by a substantial margin. - The EnsembleFusion approach attains the best overall prediction performance at 92.3%

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

### Application Domains

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

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

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

## Acknowledgements

We thank contributors and users who reported issues and shared datasets. LabelFusion builds on the open-source ecosystem, notably Hugging Face Transformers (Wolf et al., 2019), scikit-learn (Pedregosa et al., 2011), PyTorch (Paszke et al., 2019), and LLM provider SDKs. The work presented in this paper was conducted independently by the author Melchizedek Mashiku and is not affiliated with Tanaq Management Services LLC, Contracting Agency to the Division of Viral Diseases, Centers for Disease Control and Prevention, Chamblee, Georgia, USA. We acknowledge the use of the AG News and GoEmotions benchmark datasets for evaluation.

## Appendix A: Task Formalization

Formally, multi-class classification assigns each input  $x \in \mathcal{X}$  to exactly one label among  $K$  mutually exclusive classes:

$$f_{mc} : \mathcal{X} \rightarrow \{1, \dots, K\}.$$

In contrast, multi-label classification predicts a subset of relevant classes, represented as a binary indicator vector  $\mathbf{y} \in \{0, 1\}^K$ , where  $y_k = 1$  denotes membership in class  $k$ :

$$f_{ml} : \mathcal{X} \rightarrow \{0, 1\}^K.$$

This distinction shapes the training and inference stack. Multi-class models typically pair a softmax activation with categorical cross-entropy, yielding normalized class probabilities (Goodfellow et al., 2016). Multi-label classifiers instead apply independent sigmoid activations with binary cross-entropy, producing class-wise confidence scores that require calibrated thresholds at prediction time (Goodfellow et al., 2016). LabelFusion preserves these per-class semantics when concatenating transformer logits and LLM scores, allowing the fusion network to learn how much to trust each source under either regime.

## References

- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv Preprint arXiv:1810.04805*. <https://doi.org/10.48550/arXiv.1810.04805>
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. <https://doi.org/10.7551/mitpress/10251.001.0001>
- Kant, G., Wiebelt, L., Weisser, C., Kis-Katos, K., Luber, M., & Säfken, B. (2022). An iterative topic model filtering framework for short and noisy user-generated data: Analyzing conspiracy theories on twitter. *International Journal of Data Science and Analytics*, 20(2), 269–289. <https://doi.org/10.1007/s41060-022-00321-4>
- Kant, G., Zhelyazkov, I., Thielmann, A., Weisser, C., Schlee, M., Ehrling, C., Säfken, B., & Kneib, T. (2024). One-way ticket to the moon? An NLP-based insight on the phenomenon of small-scale neo-broker trading. *Social Network Analysis and Mining*, 14(1), 121. <https://doi.org/10.1007/s13278-024-01273-2>
- Lewis, D. D. (1997). Reuters-21578 text categorization test collection, distribution 1.0. *KDD Workshop on Text Mining*. <https://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html>
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv Preprint arXiv:1907.11692*. <https://doi.org/10.48550/arXiv.1907.11692>
- Luber, M., Weisser, C., Säfken, B., Silbersdorff, A., Kneib, T., & Kis-Katos, K. (2021). Identifying topical shifts in twitter streams: An integration of non-negative matrix factorisation, sentiment analysis and structural break models for large scale data. In J. Bright, A.

- 190 Giachanou, V. Spaiser, F. Spezzano, A. George, & A. Pavliuc (Eds.), *Disinformation in open*  
191 *online media* (pp. 33–49). Springer International Publishing. ISBN: 978-3-030-87031-7
- 192 OpenAI. (2023). GPT-4 technical report. *arXiv Preprint arXiv:2303.08774*. [https://doi.org/](https://doi.org/10.48550/arXiv.2303.08774)  
193 [10.48550/arXiv.2303.08774](https://doi.org/10.48550/arXiv.2303.08774)
- 194 Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin,  
195 Z., Gimelshein, N., Antiga, L., & others. (2019). PyTorch: An imperative style, high-  
196 performance deep learning library. *Advances in Neural Information Processing Systems*, 32.  
197 <https://doi.org/10.48550/arXiv.1912.01703>
- 198 Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel,  
199 M., Prettenhofer, P., Weiss, R., Dubourg, V., & others. (2011). Scikit-learn: Machine  
200 learning in python. *Journal of Machine Learning Research*, 12(Oct), 2825–2830. [http:](http://www.jmlr.org/papers/v12/pedregosa11a.html)  
201 [//www.jmlr.org/papers/v12/pedregosa11a.html](http://www.jmlr.org/papers/v12/pedregosa11a.html)
- 202 Thielmann, A. F., Weisser, C., & Säfken, B. (2024). Human in the loop: How to effectively  
203 create coherent topics by manually labeling only a few documents per class. In N. Calzolari,  
204 M.-Y. Kan, V. Hoste, A. Lenci, S. Sakti, & N. Xue (Eds.), *Proceedings of the 2024 joint*  
205 *international conference on computational linguistics, language resources and evaluation*  
206 *(LREC-COLING 2024)* (pp. 8395–8405). ELRA; ICCL. [https://doi.org/10.18653/v1/2024.](https://doi.org/10.18653/v1/2024.lrec-main.736)  
207 [lrec-main.736](https://doi.org/10.18653/v1/2024.lrec-main.736)
- 208 Thielmann, A., Weisser, C., & Krenz, A. (2021). One-class support vector machine and LDA  
209 topic model integration—evidence for AI patents. In N. H. Phuong & V. Kreinovich (Eds.),  
210 *Soft computing: Biomedical and related applications* (pp. 263–272). Springer International  
211 Publishing. [https://doi.org/10.1007/978-3-030-76620-7\\_23](https://doi.org/10.1007/978-3-030-76620-7_23)
- 212 Thielmann, A., Weisser, C., Krenz, A., & Säfken, B. (2021). Unsupervised document  
213 classification integrating web scraping, one-class SVM and LDA topic modelling. *Journal*  
214 *of Applied Statistics*, 50(3), 574–591. <https://doi.org/10.1080/02664763.2021.1919063>
- 215 Thormann, M.-L., Farchmin, J., Weisser, C., Kruse, R.-M., Säfken, B., & Silbersdorff, A.  
216 (2021). Stock price predictions with LSTM neural networks and twitter sentiment. *Statistics,*  
217 *Optimization & Information Computing*, 9(2), 268–287. [https://doi.org/10.19139/](https://doi.org/10.19139/soic-2310-5070-1202)  
218 [soic-2310-5070-1202](https://doi.org/10.19139/soic-2310-5070-1202)
- 219 Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T.,  
220 Louf, R., Funtowicz, M., & others. (2019). HuggingFace's transformers: State-of-the-art  
221 natural language processing. *arXiv Preprint arXiv:1910.03771*. [https://doi.org/10.48550/](https://doi.org/10.48550/arXiv.1910.03771)  
222 [arXiv.1910.03771](https://doi.org/10.48550/arXiv.1910.03771)
- 223 Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text  
224 classification. *Advances in Neural Information Processing Systems*, 28, 649–657. [https:](https://doi.org/10.48550/arXiv.1509.01626)  
225 [//doi.org/10.48550/arXiv.1509.01626](https://doi.org/10.48550/arXiv.1509.01626)