The Journal of Open Source Software

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

Michael Schlee[1], **Christoph Weisser**[1], **Timo Kivimäki**[2], **Melchizedek Mashiku**[4], **and Benjamin Saefken**[3]

1 Centre for Statistics, Georg-August-Universität Göttingen, Germany **2** Department of Politics and International Studies, University of Bath, Bath, UK **3** Institute of Mathematics, Clausthal University of Technology, Clausthal-Zellerfeld, Germany **4** 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

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

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 traditional transformer-based classifiers, 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 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.

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

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 via prompting. Scores can be cached to minimize API calls when cache locations are provided.
- Fusion component: a compact MLP concatenates ML logits 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)

```python
from textclassify import AutoFusionClassifier

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

clf = AutoFusionClassifier(config)
clf.fit(train_dataframe)            # trains ML backbone, gathers LLM scores, fits fus
pred = clf.predict(["This is amazing!"])  # fused prediction
```

CLI and Configuration

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

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

The Journal of Open Source Software

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%) - **GoEmotions** (Demszky et al., 2020): 28-class multi-label emotion classification for validating multi-label fusion performance

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/DataandAIReseach/LabelFusion. The fusion components require Python 3.8+ and common scientific Python dependencies (PyTorch, transformers, scikit-learn, numpy, pandas, PyYAML). Optional plotting depends on matplotlib/seaborn. Installation and quick-start snippets are provided in the README and `FUSION_README.md`.

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)	**Fusion**	**92.2%**	**0.922**	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)	**Fusion**	**92.2%**	**0.922**	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)	**Fusion**	**92.4%**	**0.924**	0.926	0.924

Training Data	Model	Accuracy	F1-Score	Precision	Recall
100% (4,000)	RoBERTa	92.2%	0.922	0.923	0.922
100% (4,000)	OpenAI	84.4%	0.844	0.857	0.844

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

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:

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

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:

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

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

Demszky, D., Movshovitz-Attias, D., Ko, J., Cowen, A., Nemade, G., & Ravi, S. (2020). GoEmotions: A dataset of fine-grained emotions. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 4040–4054. https://doi.org/10.18653/v1/2020.acl-main.372

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

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. Giachanou, V. Spaiser, F. Spezzano, A. George, & A. Pavliuc (Eds.), *Disinformation in open online media* (pp. 33–49). Springer International Publishing. ISBN: 978-3-030-87031-7

OpenAI. (2023). GPT-4 technical report. *arXiv Preprint arXiv:2303.08774*. https://doi.org/10.48550/arXiv.2303.08774

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., & others. (2019). PyTorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, *32*. https://doi.org/10.48550/arXiv.1912.01703

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., & others. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, *12*(Oct), 2825–2830. http://www.jmlr.org/papers/v12/pedregosa11a.html

Thielmann, A. F., Weisser, C., & Säfken, B. (2024). Human in the loop: How to effectively create coherent topics by manually labeling only a few documents per class. In N. Calzolari, M.-Y. Kan, V. Hoste, A. Lenci, S. Sakti, & N. Xue (Eds.), *Proceedings of the 2024 joint international conference on computational linguistics, language resources and evaluation (LREC-COLING 2024)* (pp. 8395–8405). ELRA; ICCL. https://doi.org/10.18653/v1/2024.lrec-main.736

Thielmann, A., Weisser, C., & Krenz, A. (2021). One-class support vector machine and LDA topic model integration—evidence for AI patents. In N. H. Phuong & V. Kreinovich (Eds.), *Soft computing: Biomedical and related applications* (pp. 263–272). Springer International Publishing. https://doi.org/10.1007/978-3-030-76620-7_23

Thielmann, A., Weisser, C., Krenz, A., & Säfken, B. (2021). Unsupervised document classification integrating web scraping, one-class SVM and LDA topic modelling. *Journal of Applied Statistics*, *50*(3), 574–591. https://doi.org/10.1080/02664763.2021.1919063

Thormann, M.-L., Farchmin, J., Weisser, C., Kruse, R.-M., Säfken, B., & Silbersdorff, A. (2021). Stock price predictions with LSTM neural networks and twitter sentiment. *Statistics, Optimization &Amp; Information Computing*, *9*(2), 268–287. https://doi.org/10.19139/soic-2310-5070-1202

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., & others. (2019). HuggingFace's transformers: State-of-the-art natural language processing. *arXiv Preprint arXiv:1910.03771*. https://doi.org/10.48550/arXiv.1910.03771

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. *Advances in Neural Information Processing Systems*, *28*, 649–657. https://doi.org/10.48550/arXiv.1509.01626