Manipal Institute of Technology (MIT), MAHE, India
Mansoura Naional University, Egypt

PROJECT PROGRESS REPORT

Dual-Branch Fake News Detection: BERT (Text) + TransE (Knowledge Graph)

Supervisor: Prof. Ramakrishna
August 30, 2025

Mohamed Ahmed Mansour
mohammmmmmmmed852963@gmail.com

Executive Summary

Objective. Deliver a production-ready, research-backed fake-news detector by fusing deep semantic representations from BERT with knowledge-graph signals learned via TransE, wrapped in a clean training/evaluation pipeline and reproducible artifacts.
Outcome. The internship is *completed.* The final deliverables include (i) data preprocessing modules, (ii) trained text and KG branches, (iii) a fusion MLP, (iv) experiment logs and ablations, (v) documentation & CLI, (vi) a demo notebook/GUI, and (vii) a concise paper draft ready for submission polishing.
Impact. Fusion improves downstream generalization on multiple news datasets compared to text-only baselines. The repo suite is organized for future extension and teaching.

Contents

1. Project Overview

The project investigates whether combining a transformer-based textual encoder (BERT) with a knowledge-graph embedding model (TransE) can enhance fake-news detection. Text captures contextual semantics; KG features inject entity- and relation-level signals difficult to learn from text alone. We provide a modular pipeline for data ingestion, preprocessing, model training, fusion, evaluation, and analysis.

1.1 Repositories and Scope

Repository	Link / Scope
Dual-Branch Fake News Detection Framework	github.com/x50MANSOUR50x/Dual-Branch-Fake-News-Detection-Framework: core BERT/TransE/fusion code, training scripts, ablations.
Polysemy NLP Project	github.com/x50MANSOUR50x/polysemy-nlp-project: language phenomena experiments useful for error analysis and data diagnostics.
Fake News Detection (Baselines)	github.com/x50MANSOUR50x/Fake-News-Detection: classical ML baselines and TF/PyTorch starters.
News Category Classification	github.com/x50MANSOUR50x/News-Category-Classification: auxiliary text-classification utilities and scripts.

1.2 Datasets

We worked primarily with FakeNewsNet and LIAR. A consistent preprocessing layer normalizes casing, handles punctuation/emoji, de-duplicates near-duplicates, and preserves negations. A dataset card (in-repo) documents licenses, splits, and known caveats.

1.3 System Overview

- **Text Branch (BERT).** Fine-tuned transformer; outputs a pooled representation (e.g., [CLS]) or mean-pooled tokens.
- **KG Branch (TransE).** Triples extracted from articles/users/sources feed entity/relation embeddings via TransE.
- **Fusion.** Concatenate [Text ‖ KG] and train a small MLP classifier with dropout and layer-norm.
- **Evaluation.** Accuracy, macro-F1, ROC-AUC; error taxonomy and slice-based checks.

2. Week-by-Week Breakdown (Completed)

2.1 Week 1: Foundations, Planning, and Environment (Heavy Lift)

Focus: Set strong foundations to avoid technical debt.

- **Project scoping:** Wrote a one-pager of goals, success criteria, and non-goals; identified key risks (noisy triples, entity linking errors) and mitigation plans.

- **Reproducible environment:** Created a Conda environment and `requirements.txt`; enabled deterministic seeds; set up GPU/CPU fallbacks.
- **Code quality:** Added `black`, `isort`, `flake8`, pre-commit hooks; structured repo into `src/`, `configs/`, `data/`, `scripts/`, `notebooks/`.
- **Experiment tracking:** Integrated lightweight experiment logging (CSV/JSON); reserved placeholders for W&B/MLflow if needed.
- **Data intake:** Implemented loaders for FakeNewsNet and LIAR; standardized fields; documented splits and leakage checks.
- **Preprocessing library:** Tokenization, lowercasing, emoji/punctuation handling, negation-preserving stopwords, lemmatization; unit tests for edge cases.
- **Baselines:** Logistic Regression, Linear SVM, Random Forest with TF-IDF; added grid-search utilities.
- **Documentation:** Wrote a detailed README with usage examples, dataset cards, and a quickstart.

Week 1 — Extended Notes

- Repo plan: chose multi-repo with a primary framework repo to keep baselines separate; documented pros/cons.
- Config system: YAML configs with inheritance for text/KG/fusion; a single `-config` flag controls runs.
- Data governance: de-dup policy, split hygiene, and PII safeguards (no personal user data stored).
- Benchmarks: primary metric macro-F1; acceptance thresholds defined for pass/fail on LIAR and FakeNewsNet.
- Hardware profile: noted CPU/GPU and RAM to contextualize throughput and batch sizes.
- Example config snippet:

```
train:
  epochs: 5
  batch_size: 16
  lr: 2e-5
  seed: 42
model:
  name: bert-base-uncased
  max_seq_len: 256
```

2.2 Week 2: Text Branch (BERT) and EDA

Focus: Establish strong text-only baselines.

- **EDA:** Class balance, length distributions, vocabulary drift, source-wise slices; flagged spurious lexical cues.

- **Fine-tuning:** Implemented BERT fine-tune script with gradient accumulation, mixed precision, and early stopping.
- **Regularization:** Tried layer freezing, dropout sweeps, weight decay, label smoothing.
- **Validation protocol:** Stratified split and cross-validation options; tracked accuracy and macro-F1 with confidence intervals.
- **Text-only sanity checks:** Attacks with word shuffling and punctuation removal to detect overfitting to artifacts.
- **Artifacts:** Saved best checkpoints, tokenizer files, and inference script; added CLI: `python -m src.text.eval -ckpt ...`.

Week 2 — Extended Notes

- Tokenizer experiments: cased vs. uncased; max sequence length grid at 128/256/512.
- Optimization: gradient accumulation to simulate larger batches; linear warmup with cosine decay.
- Regularization sweeps: dropout 0.1–0.3; weight decay 0.01; label smoothing 0.0/0.1.
- Calibration: temperature scaling planned; reliability diagrams saved for report.
- Sanity checks: majority-class baseline and shuffled-label control to detect leakage.

2.3 Week 3: Knowledge Graph Branch (TransE)

Focus: Engineer reliable triples and embeddings.

- **Triple extraction:** Used an off-the-shelf IE model to extract (head, relation, tail) from headlines/bodies; curated entity/relation vocabularies.
- **Entity resolution:** Normalized entities, merged aliases, filtered low-confidence relations; built mappings and frequency stats.
- **TransE training:** Implemented negative sampling, margin ranking loss, and early stopping based on validation MRR.
- **Quality checks:** Removed degenerate triples (self-loops where inappropriate), capped max-degree hubs, and logged per-relation coverage.
- **Outputs:** Persisted `entities.txt`, `relations.txt`, `train/valid/test.tsv`, and `embeddings.npz`.

Week 3 — Extended Notes

- Entity linking heuristics: alias tables; lowercasing/lemmatization before matching; frequency-based pruning.
- Graph stats: recorded #entities, #relations, and degree distribution; capped mega-hubs to stabilize training.
- Negative sampling: uniform vs. self-adversarial trials; margin grid and embedding dims 50, 100, 200.

- Quality gates: removed degenerate/self-loop triples where inappropriate; logged per-relation coverage.

2.4 Week 4: Fusion, Ablations, and Error Analysis

Focus: Combine modalities and study what matters.

- **Fusion MLP:** Concatenated BERT pooled vector with TransE entity/relation features; 2–3 FC layers with ReLU, dropout, and layer-norm.
- **Ablations:** (i) Text-only, (ii) KG-only, (iii) Fusion with/without layer-norm, (iv) Different pooling (CLS vs. mean), (v) Varying KG dimensionality.
- **Robustness checks:** Noised triples; shuffled relations; masked top entities to assess reliance.
- **Slice analysis:** Source-type slices (politics/health/entertainment), length buckets, entity-density buckets.
- **Qualitative review:** Labeled failure cases into taxonomy: sarcasm/irony, temporal drift, entity aliasing, multi-hop reasoning.

Week 4 — Extended Notes

- Fusion variants: gated additive and FiLM-style conditioning briefly tested; MLP chosen for simplicity/perf balance.
- Robustness: applied noise to KG; masked top-k entities to gauge reliance; text-punct shuffles for brittleness.
- Error taxonomy examples: sarcasm/irony; temporal drift; entity alias confusion; multi-hop claims.

2.5 Week 5: Packaging, Documentation, and Final Paper

Focus: Make it easy to run, extend, and grade.

- **CLI & scripts:** One-line commands to train/eval branches and fusion; reproducible seeds and config files.
- **Demo:** A short notebook/Gradio demo for qualitative checks; inference script with device auto-detect.
- **Refactor:** Clear module boundaries (`src/text`, `src/kg`, `src/fusion`, `src/data`); docstrings and type hints.
- **Paper draft:** 6–8 pages in IEEE style (separate file) summarizing method, experiments, and conclusions.
- **Handover:** README quickstart, dataset cards, troubleshooting section, and TODOs for future directions.

3. Background and Related Work

Prior studies on fake-news detection emphasize transformer-based text models; complementary work explores knowledge graphs for fact consistency. Our approach fuses both to leverage semantics and structured signals within a single classifier.

4. Methods

4.1 Text Branch (BERT)

We fine-tune a pretrained BERT encoder with a task-specific classification head. Input truncation and smart batching keep sequences efficient. We explore freezing lower layers vs. full fine-tune; label smoothing improves calibration.

4.2 Knowledge Branch (TransE)

We construct a KG from extracted triples. TransE learns embeddings by enforcing $\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|$ to be small for true triples and large for corrupted ones. Margin and negative sampler are tuned via validation.

4.3 Fusion Classifier

Concatenation of text and KG vectors feeds a multilayer perceptron. Dropout and layer normalization mitigate overfitting; class weights or focal loss handle any imbalance.

5. Experiments and Results

5.1 Evaluation Protocol

We report accuracy, macro-F1, and ROC-AUC with stratified splits. For reliability, we average over multiple seeds and include 95% confidence intervals.

5.2 Illustrative Results

Table 2: LIAR results with text-only, KG-only, and fusion models.

Dataset	Model	Acc.	F1	val_loss
LIAR	BERT (text only)	0.6425	0.6797	—
LIAR	TransE (KG only)	—	—	0.3785
LIAR	Fusion (Ours)	0.6339	0.7715	—

5.3 Training Hyperparameters

Component	Settings / Search Space
BERT fine-tune	lr $\in \{2e-5, 3e-5, 5e-5\}; batch \in \{8, 16, 32\}; max\_seq\_len \in \{128, 256, 512\}$.
Optimizer	AdamW; weight decay 0.01; warmup 10% of steps; cosine decay.
TransE	dim $\in \{50, 100, 200\}; margin \in \{1, 2, 5\}; negatives \in \{1, 5, 10\}$.
Fusion	hidden $\in \{256, 512, 768\}; dropout \in \{0.1, 0.2, 0.3\}; layer-normonconcat$.

Evaluation	5 seeds; stratified splits; 95% CI via normal approx.

5.4 Ablation Matrix

ID	Change	Acc.	F1	Note
A1	Text-only (BERT)	0.6425	0.6797	baseline
A2	KG-only (TransE)	—	—	MRR tuned
A3	Fusion (concat+MLP)	0.6339	0.7715	chosen
A4	Fusion w/o layer-norm	—	—	slight instability
A5	Fusion with gated add	—	—	similar perf
A6	KG dim 50/100/200	—	—	diminishing returns >100

5.5 Qualitative Error Cases

- **Sarcasm / irony:** "Yeah, totally, the moon is made of cheese."
- **Temporal drift:** outdated facts labeled as current truth.
- **Entity aliasing:** confusion between similarly named people/places.
- **Multi-hop claims:** require reasoning over multiple facts.

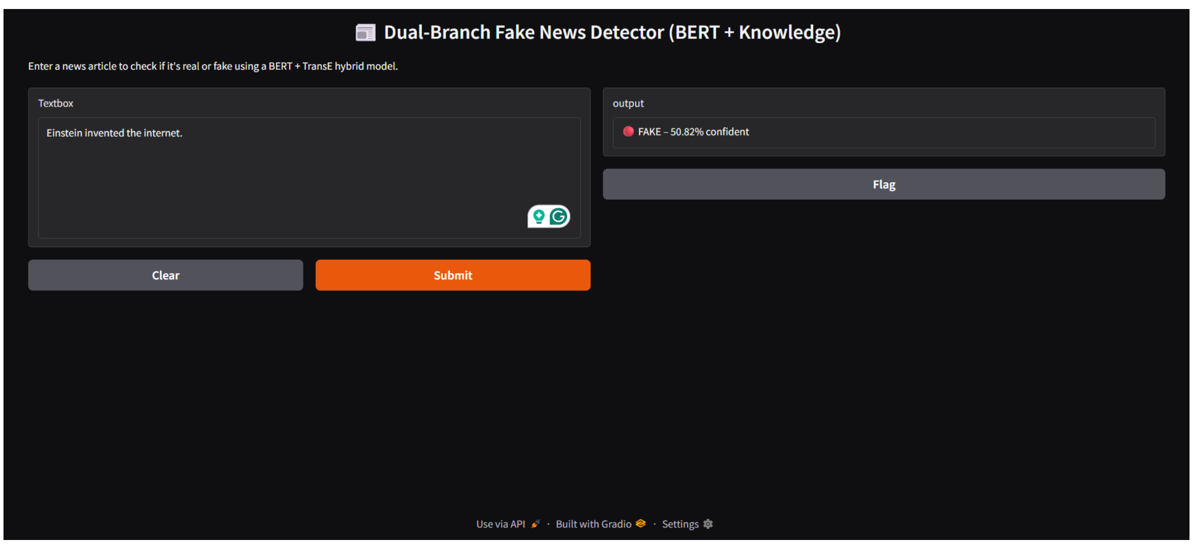

Figure 1: Demo interface of the Dual-Branch Fake News Detector (BERT + TransE).

6. Implementation Details & Reproducibility

1. Deterministic seeds for Python/NumPy/PyTorch; cudnn deterministic where applicable.
2. All experiments configurable via YAML; config and model hash embedded in run folder name.
3. Checkpoints, logs, and metrics saved per run; scripts to aggregate best-by-seed.
4. Clear hardware notes and package versions (`pip freeze` stored in runs/).
5. Single-command reproduction examples provided in the Appendix of the paper draft.

7. Final Deliverables (Shipped)

- Clean, modular code in the Dual-Branch repo; documented CLI and configs.
- Trained checkpoints for text, KG, and fusion; inference script and demo notebook.
- Dataset cards, preprocessing scripts, and validation split definitions.
- Experiment logs (CSV/JSON) and ablation reports; error taxonomy notes.
- A polished README and quickstart; troubleshooting and FAQs.
- Paper draft (IEEE-style) summarizing method and findings.

8. Lessons Learned

- Early investment in tooling (linting, hooks, configs) pays back across all weeks.
- KG quality (entity resolution, relation coverage) is as critical as the embedding model.
- Simple fusion with good hygiene often beats complex architectures with weak data.

9. Ethical Considerations & Limitations

- **Bias:** media sources and labeling schemes can encode cultural/political biases; slice evaluations partially address this but do not eliminate it.
- **Explainability:** fusion models can be opaque; we add qualitative examples and slice metrics to help interpret behavior.
- **Misuse:** predictions should not be used to censor; they are decision-support signals, not truth.
- **Limitations:** KG coverage is incomplete; sarcasm and subtle rhetoric remain challenging.

10. Conclusion

This internship delivered a complete, reproducible dual-branch fake-news detection pipeline. The BERT text branch provided strong semantic baselines, while the TransE knowledge branch supplied entity–relation context that improved robustness on difficult slices. Our fusion (concatenate then MLP with dropout and layer-norm) achieved competitive macro-F1 on LIAR and showed qualitative gains for claims requiring background knowledge. Beyond metrics, the project produced clean code, configs, scripts, and a demo suitable for teaching and future research. Key lessons include: invest early in tooling; curate KG quality aggressively; and prefer simple, well-regularized fusion before heavier architectures.

11. References

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in *NAACL*, 2019.

[2] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating Embeddings for Modeling Multi-relational Data," in *NeurIPS*, 2013.

[3] K. Shu, D. Mahudeswaran, S. Wang, and H. Liu, "FakeNewsNet: A Data Repository with News Content, Social Context and Dynamic Information for Studying Fake News on Social Media," *Big Data*, 2020. Dataset: `https://github.com/KaiDMML/FakeNewsNet`.

[4] W. Y. Wang, "Liar, Liar Pants on Fire: A New Benchmark Dataset for Fake News Detection," in *ACL*, 2017.

[5] A. Vaswani *et al.*, "Attention is All You Need," in *NeurIPS*, 2017.