IMPERIAL

Efficient Implementation of LLMs: An ECOC-based approach to reducing the vocabulary bottleneck

Harsh Rajiv Agarwal - MSc Computing (AI/ML)

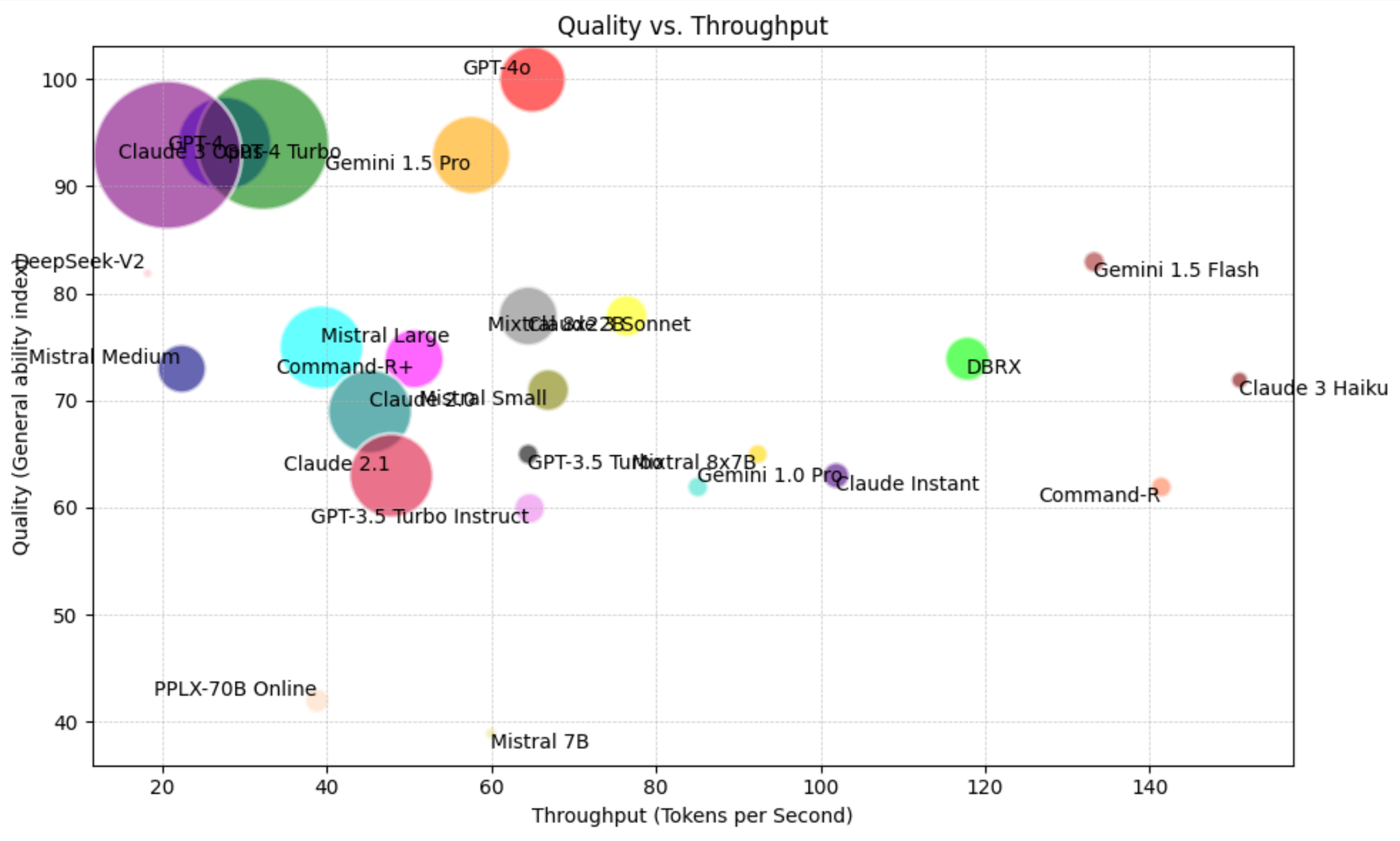
Supervisors: Dr Tolga Birdal & Dr Cemre Zor (Amazon Web Services)

13th September 2024

AGENDA

1. Motivation
2. Background - Existing Techniques & Drawbacks
3. Background - ECOC & Design Strategies
4. Proposed ECOC-based Architecture
5. Fine-tuning Setup
6. Experimental Analysis
    a. Protocol 1 – Comparison of different configurartions
    b. Protocol 2 – Effect of intermediate layer dimension
    c. Protocol 3 – Effect of different codeword length
    d. Protocol 4 – Varying different decoder backbones
7. Conclusion
8. Future Considerations

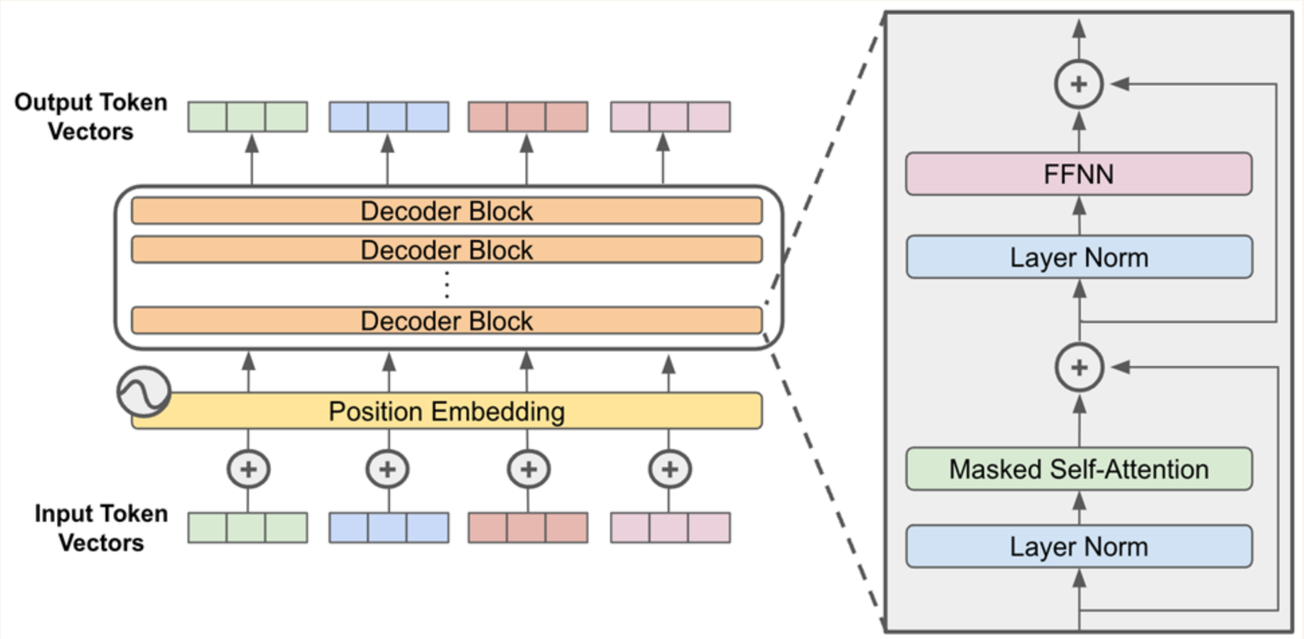
MOTIVATION



Data Source: https://artificialanalysis.ai/models

MOTIVATION

- LLMs have deep architectures comprising of:
  - Decoder layers relying on multi-head attention and feed-forward networks.
    - Time complexity is O(LSd(d + S))
  - LM head, which projects hidden states to a large vocabulary.
    - Time complexity is $O(SdV)

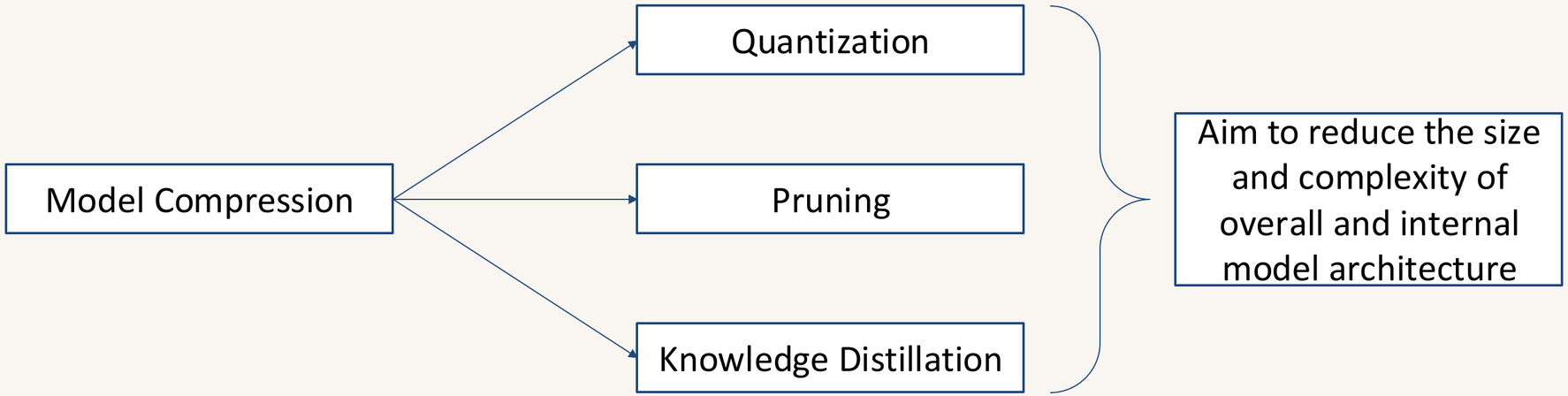


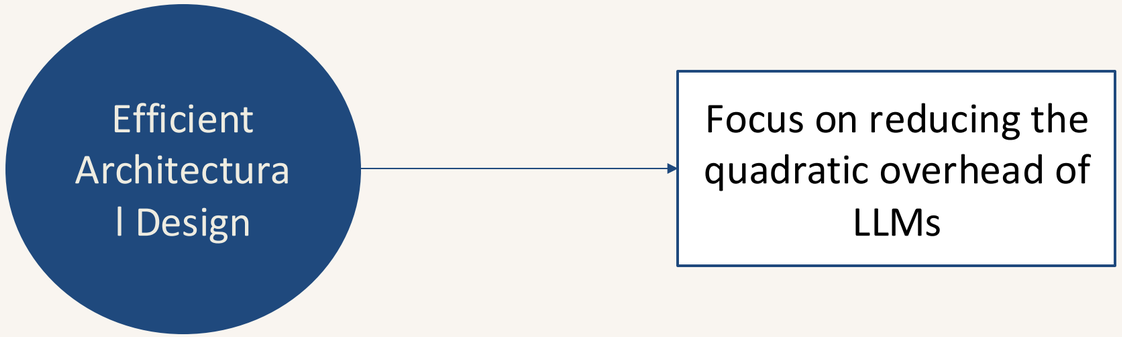
- The term “Sd” contributes significant overhead to both complexities. However, for large vocabularies V > L(d + s)

Image Source: Internet

BACKGROUND

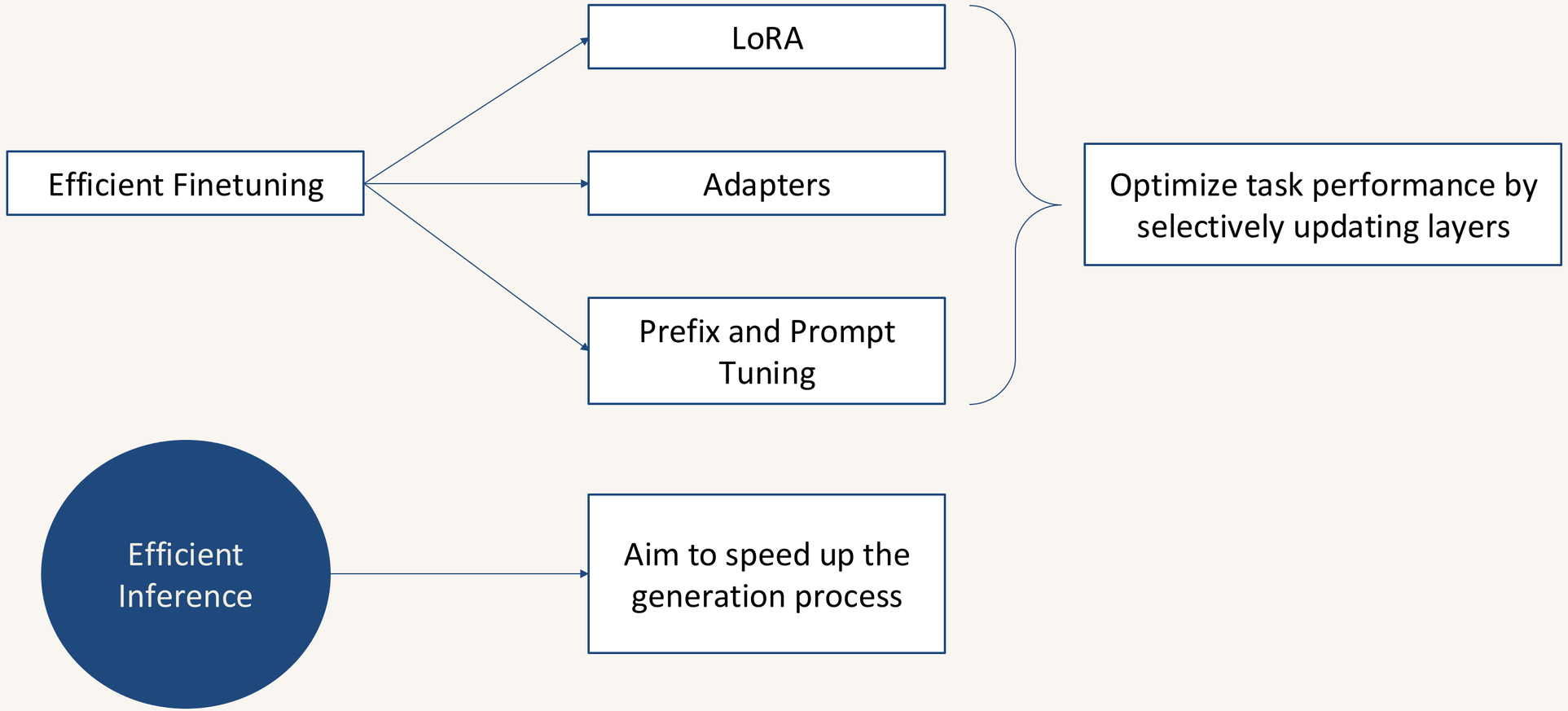
EXISTING TECHNIQUES & DRAWBACKS





BACKGROUND

EXISTING TECHNIQUES & DRAWBACKS



BACKGROUND

EXISTING TECHNIQUES & DRAWBACKS

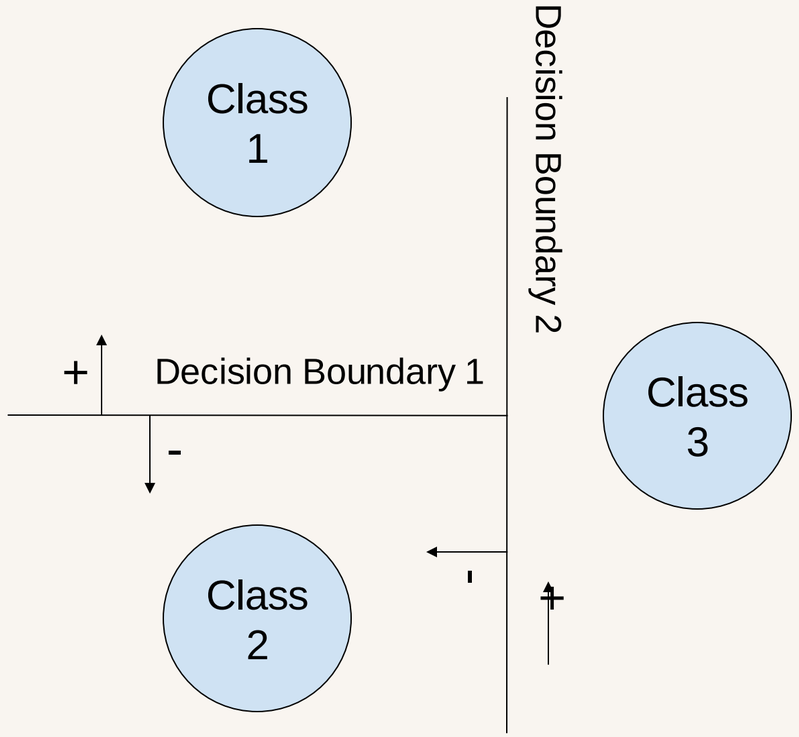
- Vocabulary Reduction: Approaches by [1] and [2] reduce vocabulary size using token frequency, semantics, and domain importance.
- Word Embedding Compression: Techniques such as K-way coding and binarization reduce embedding size by up to 98% without losing accuracy. [3-6] explore discrete code learning and binarization.
- Dynamic Compression: [7] adaptively compresses word embeddings based on task relevance using variable-length codes.

[1] Nikolay Bogoychev, Pinzhen Chen, Barry Haddow, and Alexandra Birch. 2024. The Ups and Downs of Large Language Model Inference with Vocabulary Trimming by Language Heuristics. In Proceedings of the Fifth Workshop on Insights from Negative Results in NLP. ACL.
[2] Asahi Ushio, Yi Zhou, and Jose Camacho-Collados. 2023. Efficient Multilingual Language Model Compression through Vocabulary Trimming. In Findings of the ACL: EMNLP 2023.
[3] Ting Chen, Martin Renqiang Min, and Yizhou Sun. Learning k-way d-dimensional discrete codes for compact embedding representations. In ICML 2018.
[4] Julien Tissier, Christophe Gravier, and Amaury Habrard. Near-lossless binarization of word embeddings. In Proceedings of the AAAI Conference on Artificial Intelligence, 2019.
[5] Samarth Navali, Praneet Sherki, Ramesh Inturi, and Vanraj Vala. Word embedding binarization with semantic information preservation. In Proceedings of the 28th ICCL, 2020.
[6] Siyu Liao, Jie Chen, Yanzhi Wang, Qinru Qiu, and Bo Yuan. Embedding compression with isotropic iterative quantization. In Proceedings of the AAAI Conference on Artificial Intelligence, 2020.
[7] Yeachan Kim, Kang-Min Kim, and SangKeun Lee. Adaptive compression of word embeddings. In Proceedings of the 58th annual meeting of the ACL, 2020

BACKGROUND

WHAT IS ECOC ?

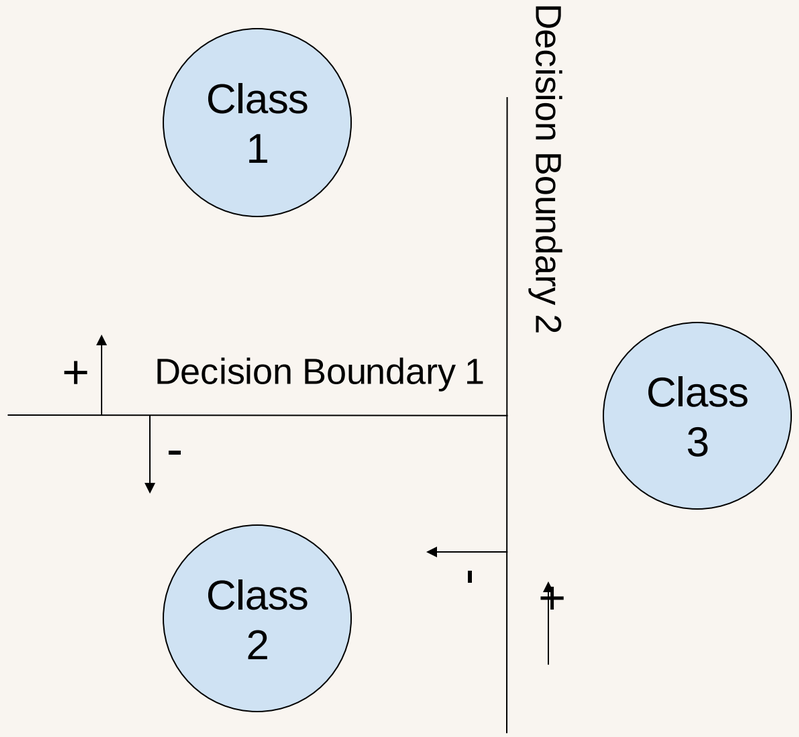
- Simplifies multi-class problems by converting them into multiple binary classification tasks through effective decision boundaries.



BACKGROUND

WHAT IS ECOC ?

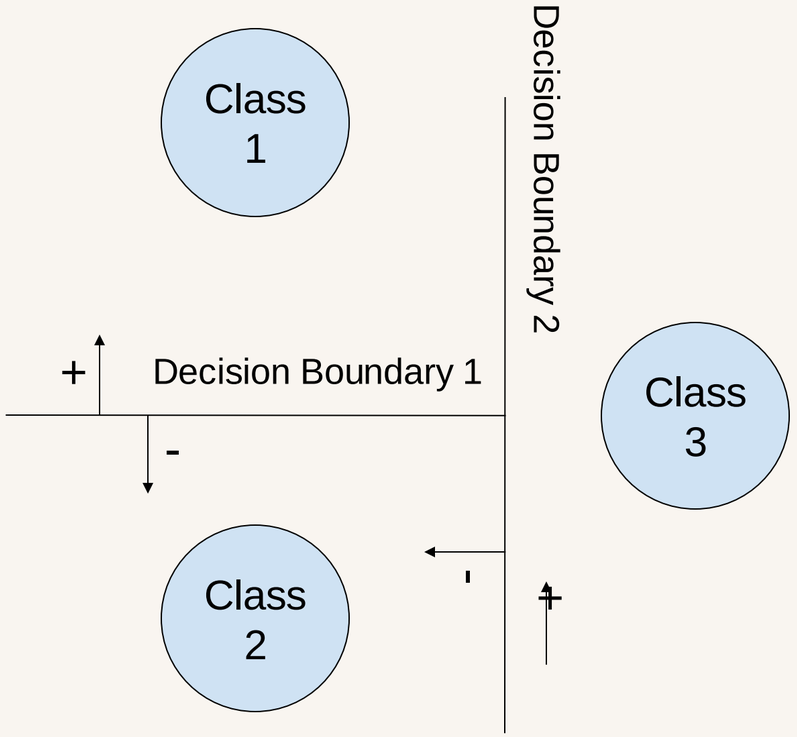
- Simplifies multi-class problems by converting them into multiple binary classification tasks through effective decision boundaries.
- Each class is represented by a unique binary or ternary codeword.



BACKGROUND

WHAT IS ECOC ?

- Simplifies multi-class problems by converting them into multiple binary classification tasks through effective decision boundaries.
- Each class is represented by a unique binary or ternary codeword.
- In codewords, each element represents the outcome of a binary classifier
  - length of a codeword = the number of binary classifiers
  - Example: [1, -1] for Class 1

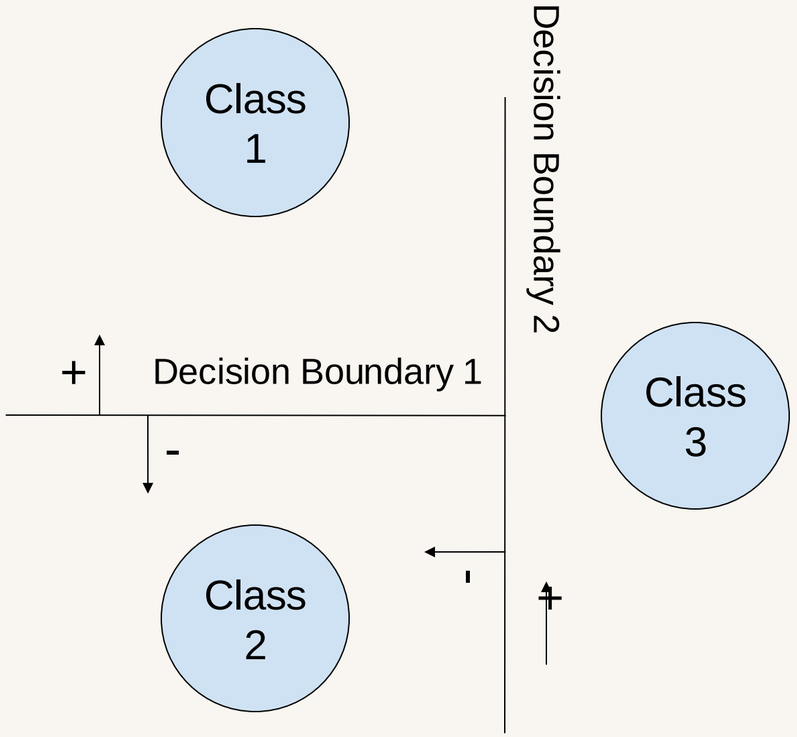


	Classifier 1	Classifier 2
Class 1	1	-1
Class 2	-1	-1
Class 3	0	1

BACKGROUND

WHAT IS ECOC ?

- Simplifies multi-class problems by converting them into multiple binary classification tasks through effective decision boundaries.
- Each class is represented by a unique binary or ternary codeword.
- In codewords, each element represents the outcome of a binary classifier
  - length of a codeword = the number of binary classifiers
  - Example: [1, -1] for Class 1
- Test sample outputs are compared to class codewords, and the class with the smallest distance (e.g., Hamming, Manhattan) is chosen.



	Classifier 1	Classifier 2
Class 1	1	-1
Class 2	-1	-1
Class 3	0	1

BACKGROUND

ECOC DESIGN STRATEGIES

Class	B^1	B^2	B^3	B^4	B^5	B^6	B^7	B^8	B^9	B^{10}
0	1	-1	-1	-1	-1	-1	-1	-1	-1	-1
1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	1	-1	-1	-1
7	-1	-1	-1	-1	-1	-1	-1	1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	1	-1
9	-1	-1	-1	-1	-1	-1	-1	-1	-1	1

One vs All (OvA)[8]

- Each classifier treats one class as positive and all others as negative
  - number of binary classifiers = number of classes

[8] Rifkin, Ryan, and Aldebaro Klautau. "In defense of one-vs-all classification." The Journal of Machine Learning Research 5 (2004): 101-141.
[9] Nilsson, N.J. (1965). Learning machines. McGrawHill: New York

BACKGROUND

ECOC DESIGN STRATEGIES

Class	B^1	B^2	B^3	B^4	B^5	B^6	B^7	B^8	B^9	B^{10}
0	1	-1	-1	-1	-1	-1	-1	-1	-1	-1
1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	1	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	1	-1	-1	-1
7	-1	-1	-1	-1	-1	-1	-1	1	-1	-1
8	-1	-1	-1	-1	-1	-1	-1	-1	1	-1
9	-1	-1	-1	-1	-1	-1	-1	-1	-1	1

One vs All (OvA)[8]

- Each classifier treats one class as positive and all others as negative
  - number of binary classifiers = number of classes

Class	B^1	B^2	B^3	B^4	B^5	B^6	B^7	B^8	B^9	B^{10}	...	B^{45}
0	1	1	1	1	1	1	1	1	1	0	...	0
1	-1	0	0	0	0	0	0	0	0	1	...	0
2	0	-1	0	0	0	0	0	0	0	0	...	1
3	0	0	-1	0	0	0	0	0	0	0	...	-1
4	0	0	0	-1	0	0	0	0	0	0	...	0
5	0	0	0	0	-1	0	0	0	0	0	...	0
6	0	0	0	0	0	-1	0	0	0	0	...	0
7	0	0	0	0	0	0	-1	0	0	0	...	0
8	0	0	0	0	0	0	0	-1	0	0	...	0
9	0	0	0	0	0	0	0	0	-1	0	...	0

One vs One (OvO)[9]

- Each classifier discriminated between a pair of classes, ignoring others
  - number of binary classifiers = c(c-1)/2, where c is the number of classes

[8] Rifkin, Ryan, and Aldebaro Klautau. "In defense of one-vs-all classification." The Journal of Machine Learning Research 5 (2004).
[9] Nilsson, N.J. (1965). Learning machines. McGrawHill: New York

BACKGROUND

ECOC DESIGN STRATEGIES

Class	B^1	B^2	B^3	B^4
0	-1	-1	-1	-1
1	-1	-1	-1	1
2	-1	-1	1	-1
3	-1	-1	1	1
4	-1	1	-1	-1
5	-1	1	-1	1
6	-1	1	1	-1
7	-1	1	1	1
8	1	-1	-1	-1
9	1	-1	-1	1

Minimal ECOC[10]

- Reduces the number of binary classifiers to $\lceil \log_2 C \rceil$, C is the number of classes
  - Consecutive class codewords differ by only one bit

[10] Miguel ´Angel Bautista, Sergio Escalera, Xavier Bar´o, Petia Radeva, Jordi Vitri`a, and Oriol Pujol. Minimal design of error-correcting output codes. Pattern Recognition Letters, 2012.
[11] Allwein, Erin L., Robert E. Schapire, and Yoram Singer. "Reducing multiclass to binary: A unifying approach for margin classifiers." Journal of machine learning research 1.Dec (2000).

BACKGROUND

ECOC DESIGN STRATEGIES

Class	B^1	B^2	B^3	B^4
0	-1	-1	-1	-1
1	-1	-1	-1	1
2	-1	-1	1	-1
3	-1	-1	1	1
4	-1	1	-1	-1
5	-1	1	-1	1
6	-1	1	1	-1
7	-1	1	1	1
8	1	-1	-1	-1
9	1	-1	-1	1

Minimal ECOC[10]

- Reduces the number of binary classifiers to $\lceil \log_2 C \rceil$, C is the number of classes
  - Consecutive class codewords differ by only one bit

Class	B^1	B^2	B^3	B^4	B^5	B^6	B^7	B^8	B^9	B^{10}	...	B^{30}
0	1	1	1	1	-1	1	1	1	1	-1	...	1
1	-1	1	-1	-1	1	-1	1	-1	1	1	...	-1
2	1	1	-1	1	1	-1	-1	1	-1	-1	...	1
3	-1	-1	1	1	-1	1	1	-1	-1	1	...	-1
4	1	-1	1	-1	1	-1	-1	1	1	-1	...	1
5	-1	1	1	1	-1	1	-1	1	-1	1	...	-1
6	1	-1	-1	1	1	-1	1	-1	1	1	...	1
7	-1	1	-1	-1	1	1	-1	1	-1	-1	...	-1
8	1	1	1	-1	-1	1	1	-1	1	-1	...	1
9	-1	-1	-1	1	1	-1	-1	1	-1	1	...	-1

Random ECOC[11]

- Each binary classifier is randomly assigned groups of classes
  - As the number of classifiers increase, optimal performance is reached

[10] Miguel ´Angel Bautista, Sergio Escalera, Xavier Bar´o, Petia Radeva, Jordi Vitri`a, and Oriol Pujol. Minimal design of error-correcting output codes. Pattern Recognition Letters, 2012.
[11] Allwein, Erin L., Robert E. Schapire, and Yoram Singer. "Reducing multiclass to binary: A unifying approach for margin classifiers." Journal of machine learning research 1.Dec (2000).

BACKGROUND

DRAWBACKS OF EXISTING DESIGN STRATEGIES

- One-vs-All (OvA):
  - Simple but creates imbalanced classifiers.
  - Less accurate decision boundaries when distinguishing one class from all others.

BACKGROUND

DRAWBACKS OF EXISTING DESIGN STRATEGIES

- One-vs-All (OvA):
  - Simple but creates imbalanced classifiers.
  - Less accurate decision boundaries when distinguishing one class from all others.
- One-vs-One (OvO):
  - Robust, but requires c(c-1)/2 classifiers, which is computationally expensive.

BACKGROUND

DRAWBACKS OF EXISTING DESIGN STRATEGIES

- One-vs-All (OvA):
  - Simple but creates imbalanced classifiers.
  - Less accurate decision boundaries when distinguishing one class from all others.
- One-vs-One (OvO):
  - Robust, but requires c(c-1)/2 classifiers, which is computationally expensive.
- Minimal ECOC:
  - Efficient with minimal classifiers, but lacks robustness.
  - Struggles when class boundaries are close.

BACKGROUND

DRAWBACKS OF EXISTING DESIGN STRATEGIES

- One-vs-All (OvA):
  - Simple but creates imbalanced classifiers.
  - Less accurate decision boundaries when distinguishing one class from all others.
- One-vs-One (OvO):
  - Robust, but requires c(c-1)/2 classifiers, which is computationally expensive.
- Minimal ECOC:
  - Efficient with minimal classifiers, but lacks robustness.
  - Struggles when class boundaries are close.
- Random ECOC:
  - Robust, but the large number of classifiers increases computational cost.

PROPOSED ECOC-BASED ARCHITECTURE

PROPOSED MINRANDOM ECOC DESIGN

- Merges compact Minimal ECOC with the variability of Random ECOC.

PROPOSED ECOC-BASED ARCHITECTURE

PROPOSED MINRANDOM ECOC DESIGN

- Merges compact Minimal ECOC with the variability of Random ECOC.
- Starts with minimal code length of$\lceil \log_2 N \rceil$, then adds R random bits to improve error resilience.
- Code length: $\lceil \log_2 N \rceil$+ R, where R adds redundancy and diversity.

Class	Minimal Bits				Random Bits			
	B^1	B^2	B^3	B^4	B^5	B^6	B^7	B^8
0	-1	-1	-1	-1	-1	1	1	-1
1	-1	-1	-1	1	-1	-1	1	1
2	-1	-1	1	-1	1	1	1	-1
3	-1	-1	1	1	1	-1	1	1
4	-1	1	-1	-1	-1	1	-1	1
5	-1	1	-1	1	-1	1	-1	-1
6	-1	1	1	-1	-1	-1	1	-1
7	-1	1	1	1	-1	-1	-1	-1
8	1	-1	-1	-1	1	-1	-1	1
9	1	-1	-1	1	1	-1	1	1

PROPOSED ECOC-BASED ARCHITECTURE

PROPOSED MINRANDOM ECOC DESIGN

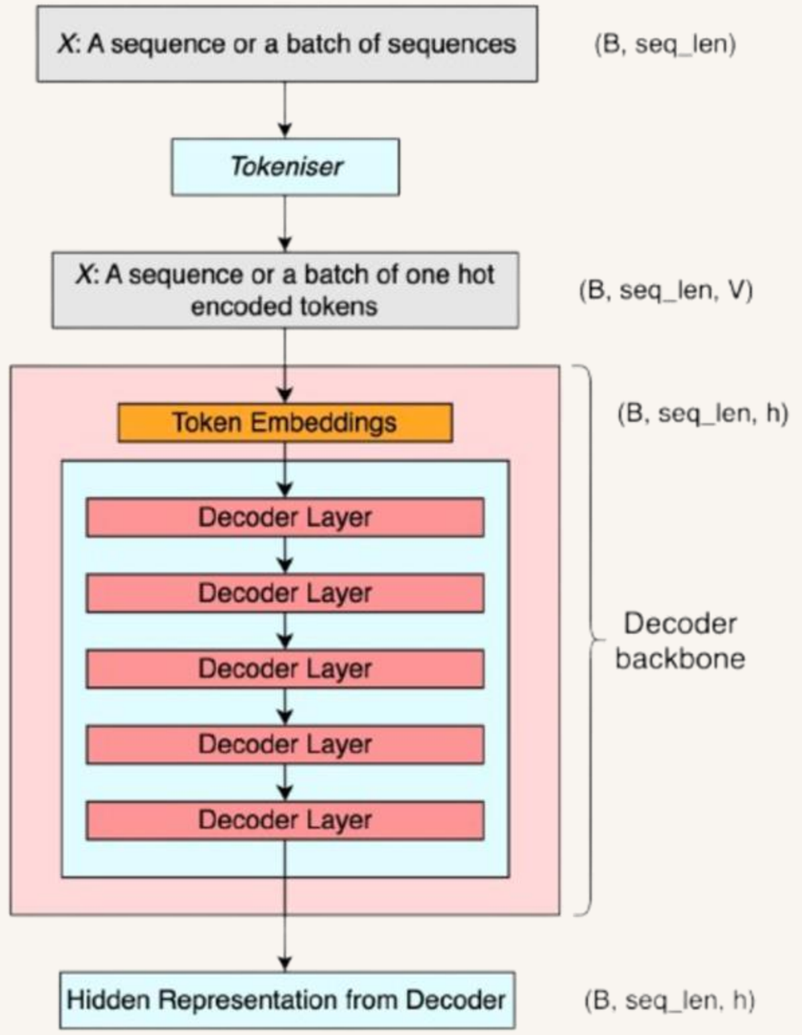
- Merges compact Minimal ECOC with the variability of Random ECOC.
- Starts with minimal code length of$\lceil \log_2 N \rceil$, then adds R random bits to improve error resilience.
- Code length: $\lceil \log_2 N \rceil$+ R, where R adds redundancy and diversity.
- Enhances robustness with a minimal increase in computational overhead.

Class	Minimal Bits				Random Bits			
	B^1	B^2	B^3	B^4	B^5	B^6	B^7	B^8
0	-1	-1	-1	-1	-1	1	1	-1
1	-1	-1	-1	1	-1	-1	1	1
2	-1	-1	1	-1	1	1	1	-1
3	-1	-1	1	1	1	-1	1	1
4	-1	1	-1	-1	-1	1	-1	1
5	-1	1	-1	1	-1	1	-1	-1
6	-1	1	1	-1	-1	-1	1	-1
7	-1	1	1	1	-1	-1	-1	-1
8	1	-1	-1	-1	1	-1	-1	1
9	1	-1	-1	1	1	-1	1	1

PROPOSED ECOC-BASED ARCHITECTURE

DECODER BACKBONE

- Input sequences are tokenized and transformed into dense, continuous vectors through an embedding layer.
- Utilizes Transformer architecture with self-attention and feed-forward operations.
- Enriched token representation is passed to the ECOC head by capturing contextual dependencies.
- OPT decoder[12] is chosen due to its versatile design that allows scaling and finetuning based on resources.

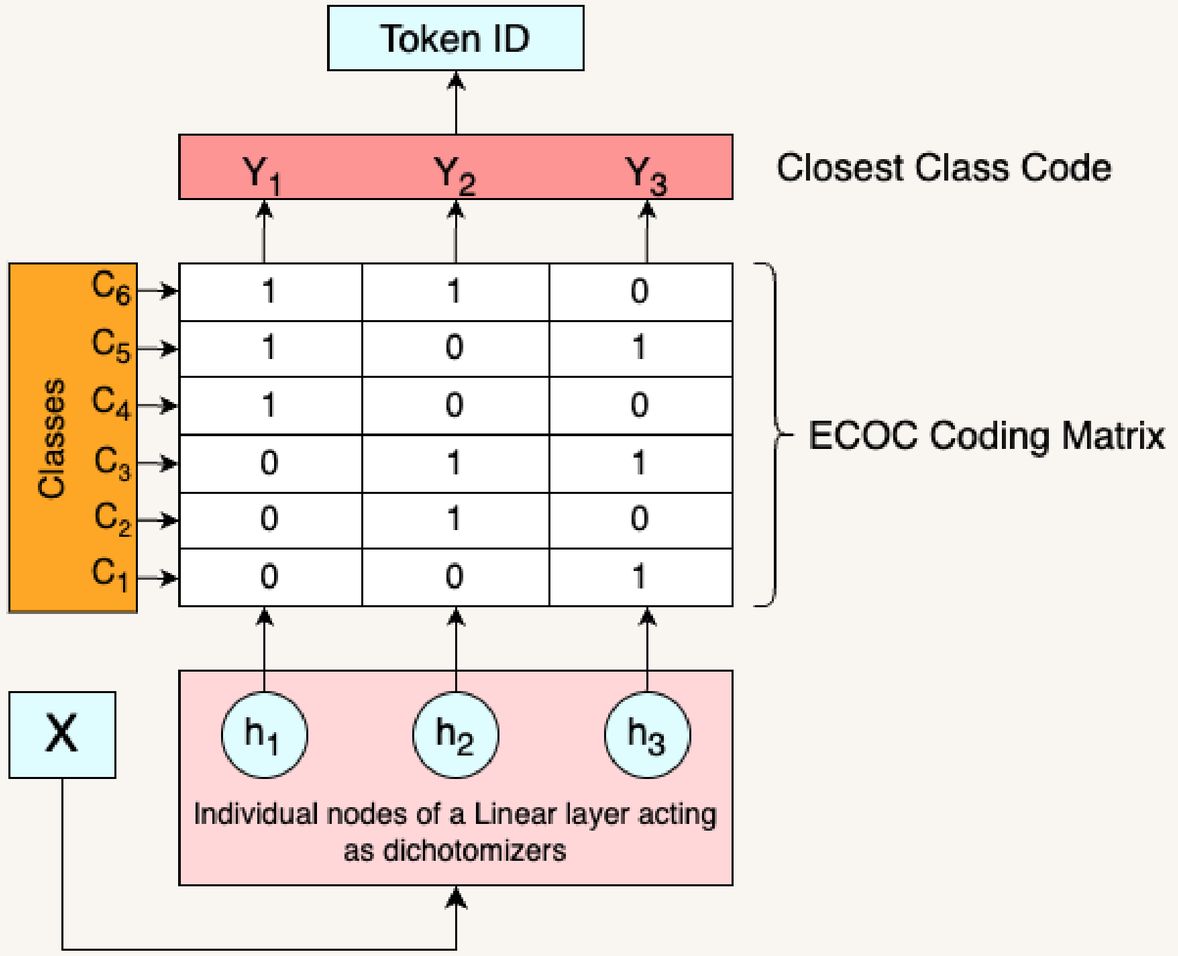


[12] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022. URL https://arxiv.org/abs/2205.01068. pages 24, 34

PROPOSED ECOC-BASED ARCHITECTURE

OVERVIEW OF ECOC BASED HEAD

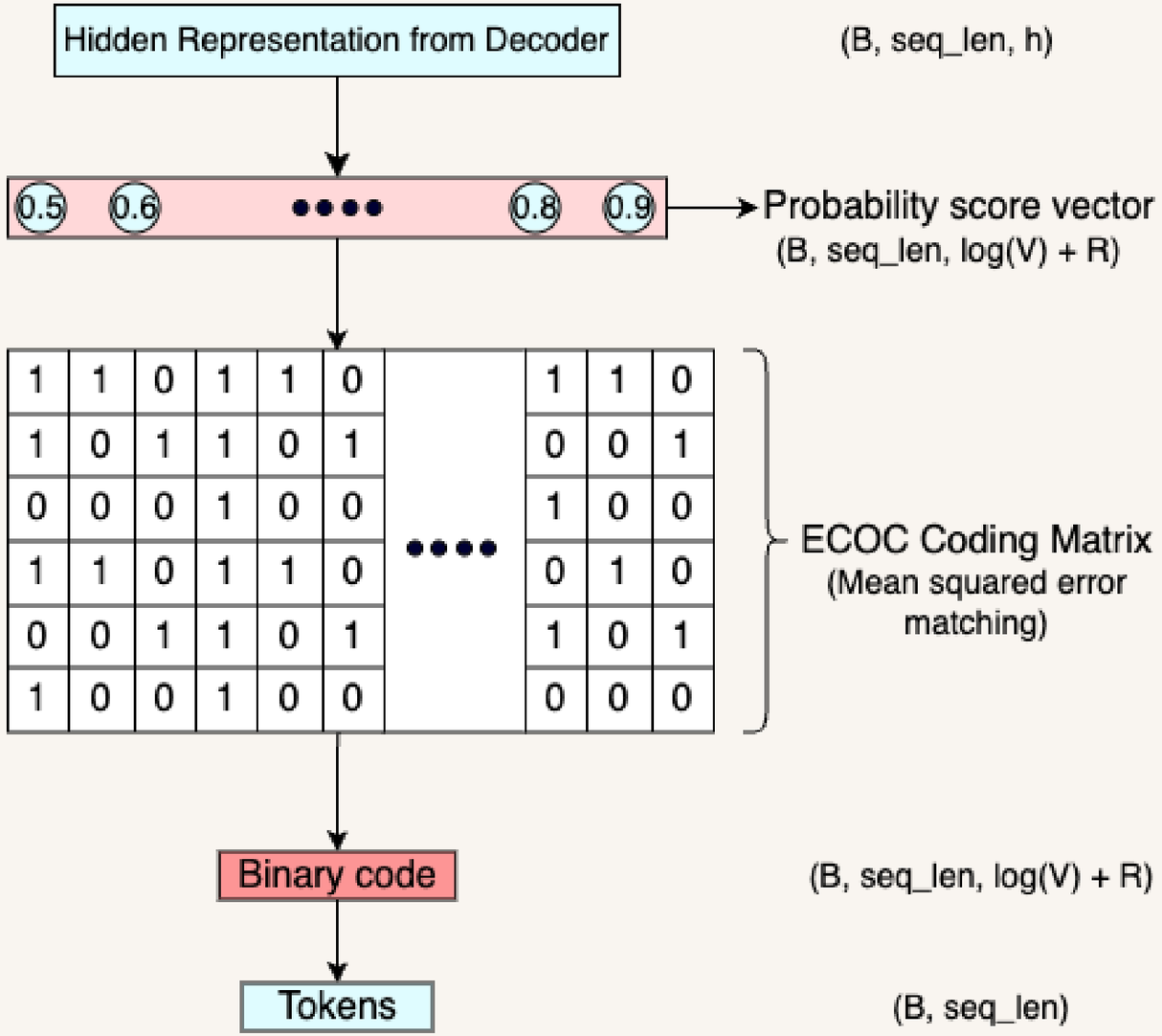
- ECOC-based head reduces decoding complexity by using binary codewords, with each node acting as a binary classifier (dichotomizer).
- Dichotomizers output soft labels, forming a prediction vector.
- Prediction vector is matched to the closest binary codeword using a distance metric, then decoded into the final Token ID.
- Output layer complexity is reduced from O(V) to O(logV).



PROPOSED ECOC-BASED ARCHITECTURE

PROPOSED DIFFERENT ECOC BASED HEADS

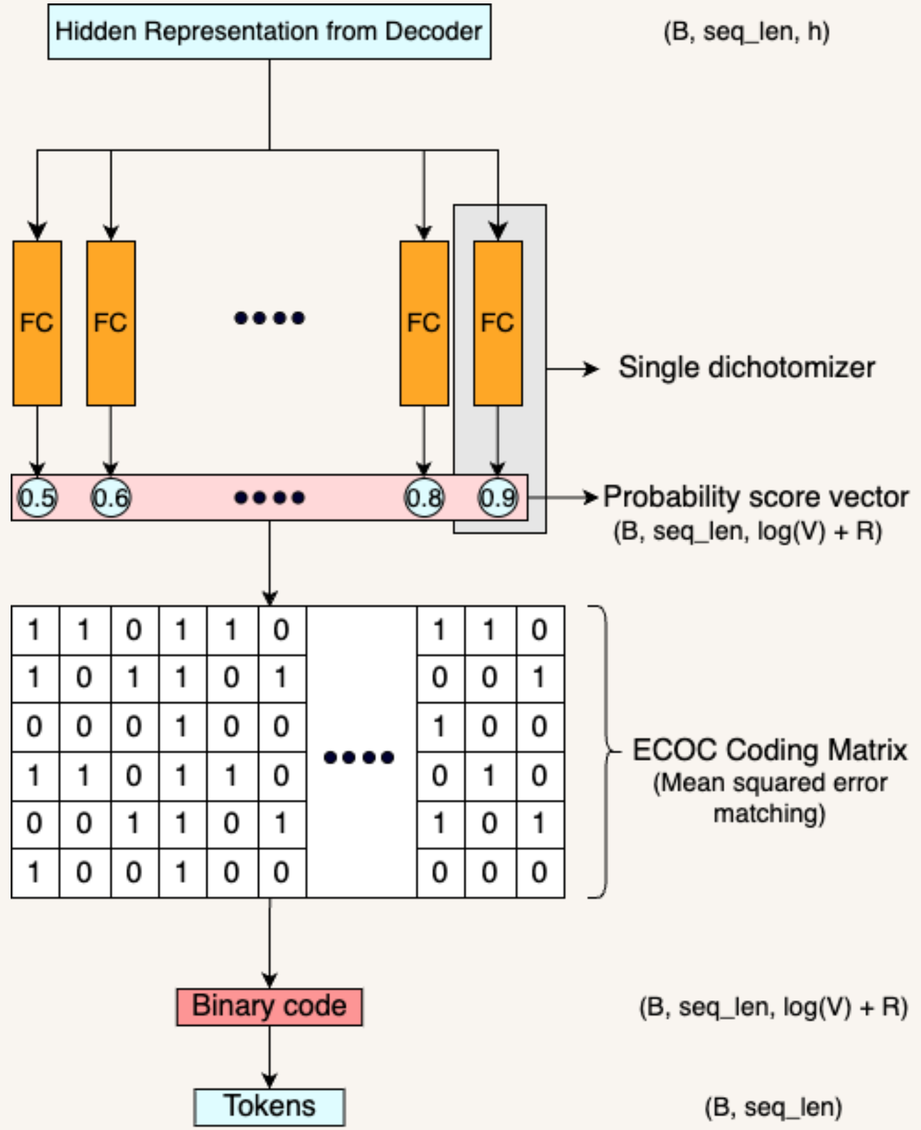
- Replaces traditional large output layer with a compact structure.
- Decoder output is passed through the ECOC head, where each node predicts the probability between 0 and 1 by a sigmoid function for that bit position
- Prediction vector is matched to the closest binary codeword using a distance metric, then decoded into the final Token ID.



PROPOSED ECOC-BASED ARCHITECTURE

PROPOSED DIFFERENT ECOC BASED HEADS

- Replaces traditional large output layer with a multi-task learning structure.
- Before each node in the ECOC layer, an additional FC layer refines the decoder embeddings for more nuanced feature extraction.
- Output from each FC layer is fed into a node that generates a probability between 0 and 1 using a sigmoid activation.



FINE-TUNING SETUP

- The combined architecture was finetuned on the Stanford Alpaca[13] dataset
  - Focus on mainly the attention weights and LM head
  - tuned on the next token prediction task

- Loss Function: Binary cross entropy loss for i^{th} bit.

$$\mathcal{L}_i = -\left(y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)\right)$$

- Evaluation Metrics:
  - BERT-Precision[14]
  - BERT-Recall[14]
  - BERT-F1[14]

[13] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.
[14] Zhang, Tianyi, et al. "Bertscore: Evaluating text generation with bert." arXiv preprint arXiv:1904.09675 (2019).

EXPERIMENTAL ANALYSIS

PROTOCOL 1: COMPARISON OF DIFFERENT CONFIGURATIONS

- Traditional LM Head:
  - High computational burden due to large parameter count and softmax complexity.

Configuration	Number of parameters in last layer	Inference Time (in mins:secs)
Without ECOC	**102,957,056**	**12:50**
Minimal ECOC (16, 0)	**32,768**	**01:05**
MinRandom ECOC (50, 0)	102,400	01:45
Minimal MTL-ECOC (16, 512)	**16,785,408**	**07:00**
MinRandom MTL-ECOC (50, 512)	52,454,400	08:10

EXPERIMENTAL ANALYSIS

PROTOCOL 1: COMPARISON OF DIFFERENT CONFIGURATIONS

- Traditional LM Head:
  - High computational burden due to large parameter count and softmax complexity.
- Minimal ECOC & MinRandom ECOC:
  - Highly efficient but leads to considerable loss in F1 score.

Configuration	Number of parameters in last layer	Inference Time (in mins:secs)
Without ECOC	**102,957,056**	**12:50**
Minimal ECOC (16, 0)	**32,768**	**01:05**
MinRandom ECOC (50, 0)	102,400	01:45
Minimal MTL-ECOC (16, 512)	**16,785,408**	**07:00**
MinRandom MTL-ECOC (50, 512)	52,454,400	08:10

EXPERIMENTAL ANALYSIS

PROTOCOL 1: COMPARISON OF DIFFERENT CONFIGURATIONS

- Traditional LM Head:
  - High computational burden due to large parameter count and softmax complexity.
- Minimal ECOC & MinRandom ECOC:
  - Highly efficient but leads to considerable loss in F1 score.
- MTL-ECOC:
  - Balances parameter reduction with improved accuracy.
  - Random bits increase parameters for enhanced robustness.

Configuration	Number of parameters in last layer	Inference Time (in mins:secs)
Without ECOC	**102,957,056**	**12:50**
Minimal ECOC (16, 0)	**32,768**	**01:05**
MinRandom ECOC (50, 0)	102,400	01:45
Minimal MTL-ECOC (16, 512)	**16,785,408**	**07:00**
MinRandom MTL-ECOC (50, 512)	52,454,400	08:10

EXPERIMENTAL ANALYSIS

PROTOCOL 1: COMPARISON OF DIFFERENT CONFIGURATIONS

- Minimal & MinRandom ECOC:
  - Observation: Higher training losses with fluctuations.
  - Challenge: Instability due to noisy bit predictions and insufficient capacity.

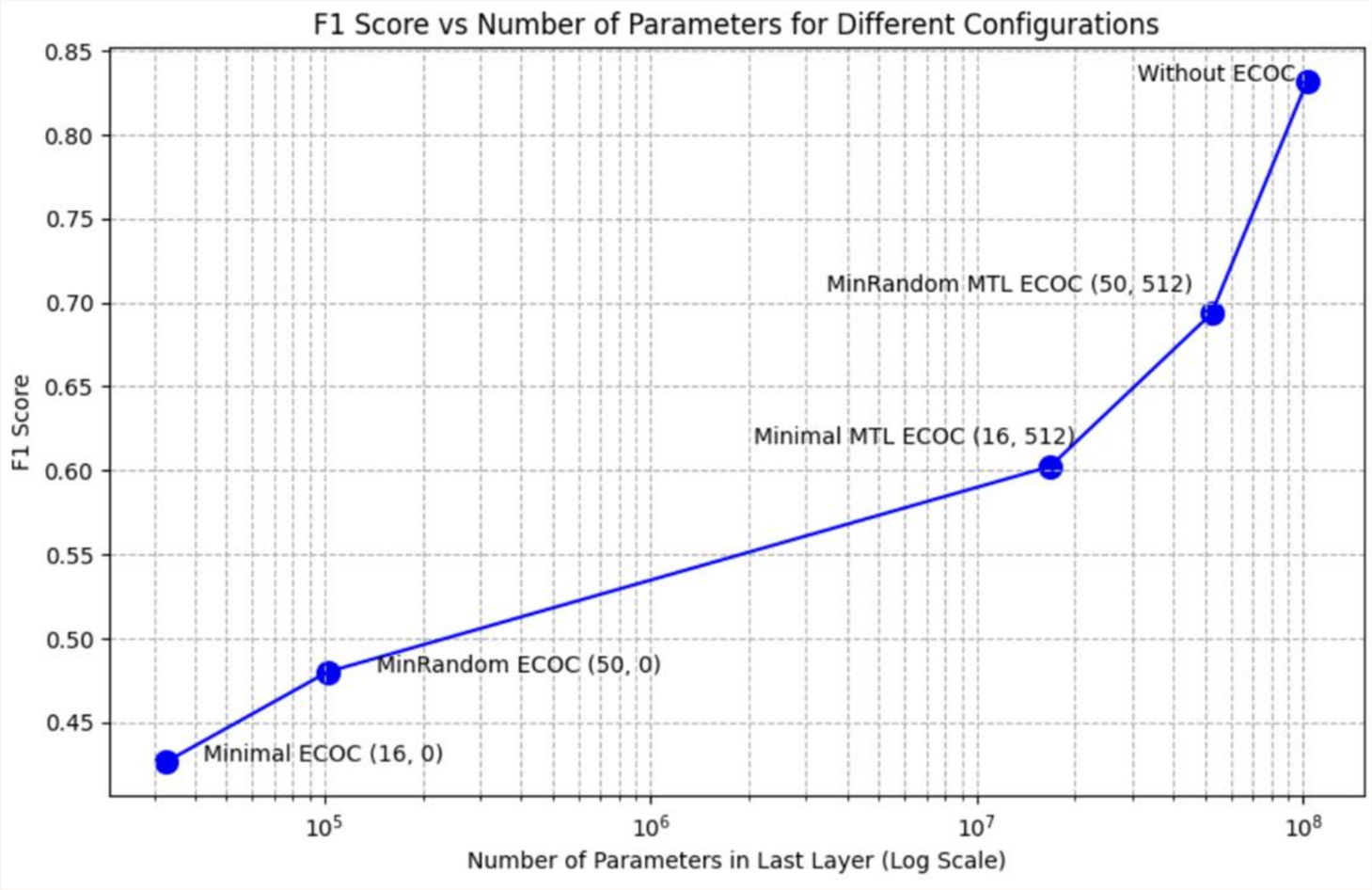
EXPERIMENTAL ANALYSIS

PROTOCOL 1: COMPARISON OF DIFFERENT CONFIGURATIONS

- Minimal & MinRandom ECOC:
  - Observation: Higher training losses with fluctuations.
  - Challenge: Instability due to noisy bit predictions and insufficient capacity.
- MTL-ECOC:
  - Observation: Lower, more stable training losses from early stages.
  - Benefit: Added layer in MTL configurations aids fine-tuning and quicker convergence.

EXPERIMENTAL ANALYSIS

PROTOCOL 1: OBSERVED TRADEOFF



EXPERIMENTAL ANALYSIS

PROTOCOL 2: EFFECT OF INTERMEDIATE LAYER DIMENSION IN MTL-ECOC

- Increasing dimension size improves performance for all MTL-ECOC configurations.

Configuration	Final Mean Training Loss per node	Precision	Recall	F1 Score
Minimal MTL-ECOC (16, 1024)	0.2305	0.6023	0.6047	0.6034
Minimal MTL-ECOC (16, 512)	0.2464	0.5924	0.6126	0.6023
Minimal MTL-ECOC (16, 128)	0.2657	0.5749	0.5925	0.5835
MinRandom MTL-ECOC (50, 1024)	**0.2829**	**0.6648**	**0.7439**	**0.7014**
MinRandom MTL-ECOC (50, 512)	0.3134	0.6577	0.7348	0.6934
MinRandom MTL-ECOC (50, 128)	0.3230	0.6256	0.7078	0.6641

EXPERIMENTAL ANALYSIS

PROTOCOL 2: EFFECT OF INTERMEDIATE LAYER DIMENSION IN MTL-ECOC

- Increasing dimension size improves performance for all MTL-ECOC configurations.
- Performance Saturation:
  - F1 score improvement plateaus beyond 512 nodes.
- Smaller dimensions indicate insufficient representational power

Configuration	Final Mean Training Loss per node	Precision	Recall	F1 Score
Minimal MTL-ECOC (16, 1024)	0.2305	0.6023	0.6047	0.6034
Minimal MTL-ECOC (16, 512)	0.2464	0.5924	0.6126	0.6023
Minimal MTL-ECOC (16, 128)	0.2657	0.5749	0.5925	0.5835
MinRandom MTL-ECOC (50, 1024)	**0.2829**	**0.6648**	**0.7439**	**0.7014**
MinRandom MTL-ECOC (50, 512)	0.3134	0.6577	0.7348	0.6934
MinRandom MTL-ECOC (50, 128)	0.3230	0.6256	0.7078	0.6641

EXPERIMENTAL ANALYSIS

PROTOCOL 2: EFFECT OF INTERMEDIATE LAYER DIMENSION IN MTL-ECOC

- Increasing dimension size improves performance for all MTL-ECOC configurations.
- Performance Saturation:
  - F1 score improvement plateaus beyond 512 nodes.
- Smaller dimensions indicate insufficient representational power
- MinRandom MTL-ECOC outperforms Minimal MTL-ECOC

Configuration	Final Mean Training Loss per node	Precision	Recall	F1 Score
Minimal MTL-ECOC (16, 1024)	0.2305	0.6023	0.6047	0.6034
Minimal MTL-ECOC (16, 512)	0.2464	0.5924	0.6126	0.6023
Minimal MTL-ECOC (16, 128)	0.2657	0.5749	0.5925	0.5835
MinRandom MTL-ECOC (50, 1024)	**0.2829**	**0.6648**	**0.7439**	**0.7014**
MinRandom MTL-ECOC (50, 512)	0.3134	0.6577	0.7348	0.6934
MinRandom MTL-ECOC (50, 128)	0.3230	0.6256	0.7078	0.6641

EXPERIMENTAL ANALYSIS

PROTOCOL 3: EFFECT OF DIFFERENT CODEWORD LENGTH

- Increasing codeword length improves performance for all MTL-ECOC configurations.
- Performance Saturation:
  - F1 score improvement plateaus after a certain codeword length.

Configuration	Final Mean Training Loss per node	Precision	Recall	F1 Score
Minimal MTL-ECOC (16, 512)	0.2464	0.5924	0.6126	0.6023
MinRandom MTL-ECOC (25, 512)	0.2760	0.6359	0.6583	0.6469
MinRandom MTL-ECOC (50, 512)	0.3134	0.6577	0.7348	0.6934
MinRandom MTL-ECOC (75, 512)	0.3177	0.6689	0.7278	0.6971
MinRandom MTL-ECOC (100, 512)	0.2583	0.6835	0.7551	0.7175

EXPERIMENTAL ANALYSIS

PROTOCOL 4: VARYING DIFFERENT DECODER BACKBONES

- Best performance achieved by Qwen2-1.5B
  - Advanced optimizations like GQA and DCA improve handling of longer contexts and complex tasks

Configuration	Final Mean Training Loss per node	Precision	Recall	F1 Score
OPT-1.3B MinRandom MTL-ECOC (50, 512)	0.3134	0.6577	0.7348	0.6934
Qwen2-1.5B MinRandom MTL-ECOC (75, 512)	**0.2821**	**0.6851**	**0.7759**	**0.7268**
TinyLlama-1.1B MinRandom MTL-ECOC (100, 512)	0.2693	0.66835	0.7579	0.7102

CONCLUSION

- First comprehensive effort to reduce complexity in the LM head layer, unlike previous work focused on model size reduction or attention mechanism efficiency.

CONCLUSION

- First comprehensive effort to reduce complexity in the LM head layer, unlike previous work focused on model size reduction or attention mechanism efficiency.
- ECOC framework reduces the LM head size to 15% of its original capacity while achieving up to 75% of the original F1 score.
- Significant decrease in inference time while maintaining competitive performance.

CONCLUSION

- First comprehensive effort to reduce complexity in the LM head layer, unlike previous work focused on model size reduction or attention mechanism efficiency.
- ECOC framework reduces the LM head size to 15% of its original capacity while achieving up to 75% of the original F1 score.
- Significant decrease in inference time while maintaining competitive performance.
- Experimental Findings:
  - Tradeoff: Increased model complexity leads to improved predictive accuracy.
  - Adding random bits improves generalization and robustness.
  - Optimizing intermediate layer size in MTL-ECOC enhances precision and recall.
  - Enhancing codeword length is better than increasing intermediate layer size in MTL-ECOC.

FUTURE CONSIDERATIONS

- Problem-Dependent ECOC Designs:
  - Explore techniques like DECOC[15], Forest-ECOC[16], and ECOC-ONE[17] to dynamically learn decision boundary partitions and generate task-specific codewords.
  - Optimize the trade-off between efficiency and accuracy for better task adaptation.

[15] Oriol Pujol, Petia Radeva, and Jordi Vitria. Discriminant ecoc: A heuristic method for application dependent design of error correcting output codes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(6):1007–1012, 2006.
[16] Sergio Escalera, Oriol Pujol, and Petia Radeva. Boosted landmarks of contextual descriptors and forest-ecoc: A novel framework to detect and classify objects in cluttered scenes. Pattern Recognition Letters, 28(13):1759–1768, 2007.
[17] Oriol Pujol, Sergio Escalera, and Petia Radeva. An incremental node embedding technique for error correcting output codes. Pattern Recognition, 41(2):713–725, 2008.

FUTURE CONSIDERATIONS

- Problem-Dependent ECOC Designs:
  - Explore techniques like DECOC[15], Forest-ECOC[16], and ECOC-ONE[17] to dynamically learn decision boundary partitions and generate task-specific codewords.
  - Optimize the trade-off between efficiency and accuracy for better task adaptation.
- Incorporate token dependencies into binary encoding to enhance semantic understanding.

[15] Oriol Pujol, Petia Radeva, and Jordi Vitria. Discriminant ecoc: A heuristic method for application dependent design of error correcting output codes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(6):1007–1012, 2006.
[16] Sergio Escalera, Oriol Pujol, and Petia Radeva. Boosted landmarks of contextual descriptors and forest-ecoc: A novel framework to detect and classify objects in cluttered scenes. Pattern Recognition Letters, 28(13):1759–1768, 2007.
[17] Oriol Pujol, Sergio Escalera, and Petia Radeva. An incremental node embedding technique for error correcting output codes. Pattern Recognition, 41(2):713–725, 2008.

FUTURE CONSIDERATIONS

- Problem-Dependent ECOC Designs:
  - Explore techniques like DECOC[15], Forest-ECOC[16], and ECOC-ONE[17] to dynamically learn decision boundary partitions and generate task-specific codewords.
  - Optimize the trade-off between efficiency and accuracy for better task adaptation.
- Incorporate token dependencies into binary encoding to enhance semantic understanding.
- Jointly optimize ECOC head and model during training to better adapt weights to task-specific data, improving accuracy.

[15] Oriol Pujol, Petia Radeva, and Jordi Vitria. Discriminant ecoc: A heuristic method for application dependent design of error correcting output codes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(6):1007–1012, 2006.
[16] Sergio Escalera, Oriol Pujol, and Petia Radeva. Boosted landmarks of contextual descriptors and forest-ecoc: A novel framework to detect and classify objects in cluttered scenes. Pattern Recognition Letters, 28(13):1759–1768, 2007.
[17] Oriol Pujol, Sergio Escalera, and Petia Radeva. An incremental node embedding technique for error correcting output codes. Pattern Recognition, 41(2):713–725, 2008.

FUTURE CONSIDERATIONS

- Problem-Dependent ECOC Designs:
  - Explore techniques like DECOC[15], Forest-ECOC[16], and ECOC-ONE[17] to dynamically learn decision boundary partitions and generate task-specific codewords.
  - Optimize the trade-off between efficiency and accuracy for better task adaptation.
- Incorporate token dependencies into binary encoding to enhance semantic understanding.
- Jointly optimize ECOC head and model during training to better adapt weights to task-specific data, improving accuracy.
- ECOC is architecture-agnostic and can be extended to multi-modal models, improving performance in tasks with diverse output spaces.

[15] Oriol Pujol, Petia Radeva, and Jordi Vitria. Discriminant ecoc: A heuristic method for application dependent design of error correcting output codes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(6):1007–1012, 2006.
[16] Sergio Escalera, Oriol Pujol, and Petia Radeva. Boosted landmarks of contextual descriptors and forest-ecoc: A novel framework to detect and classify objects in cluttered scenes. Pattern Recognition Letters, 28(13):1759–1768, 2007.
[17] Oriol Pujol, Sergio Escalera, and Petia Radeva. An incremental node embedding technique for error correcting output codes. Pattern Recognition, 41(2):713–725, 2008.

THANKS !!