DART: DYNAMIC ATTENTION-BASED REINFORCED TAU FOR ADAPTIVE REPRESENTATION LEARNING

Praneeth Kumar Palepu
AIML Team

***Abstract*— We introduce DART, a novel framework that learns to adaptively fuse multiple embeddings using a mathematically grounded controller known as tau. DART dynamically assigns importance to different representations per instance by smart embedding fusion techniques. Our framework is motivated by the observation that full finetuning of large models like BERT is both resource-intensive and environmentally unsustainable. DART achieves strong performance on sentiment classification (91.4% on SST-2) while training only ~440K parameters, offering a compelling alternative to 110M+ parameter transformer finetuning.**

I. INTRODUCTION (*HEADING 1*)

The past decade has seen a revolution in natural language processing, led by the advent of large-scale transformer models such as BERT, GPT, and their derivatives ([2], [3], [4]). While these models have achieved state-of-the-art results across many tasks, they come with substantial limitations in terms of computational cost, environmental impact, and deployment feasibility. BERT-base (uncased), for example, has approximately 110 million parameters. Finetuning such a model for a single downstream task like sentiment classification incurs significant energy consumption. A study by Strubell et al. [1] estimated that training a single transformer model can emit as much as 626,000 pounds of CO_2—equivalent to the lifetime emissions of five average cars. When gains in task accuracy are marginal (e.g., improving accuracy by 1–2%), this cost-benefit trade-off becomes questionable. Moreover, full finetuning requires updating all parameters, which is often redundant for applications with constrained data, low-latency requirements, or real-time inference needs. The one-size-fits-all nature of such models overlooks the diversity of linguistic patterns in real-world inputs. A potential remedy is selective fusion: combining the strengths of multiple pretrained embeddings—each encoding different semantic or syntactic information (e.g., FastText for local context, MPNet for global semantics, TF-IDF for frequency-based salience). However, naive concatenation leads to overparameterization and suboptimal performance. We propose that a smart, adaptive fusion mechanism can generate new embeddings that are tailored to the specific input instance. These embeddings combine multiple views of data and yield competitive accuracy, while training only a fraction of the parameters. This results in lower memory usage, faster convergence, and significantly reduced carbon footprint. DART offers a principled framework for such a fusion process. By learning a controller τ over multiple projected embeddings using a mathematically grounded formulation, DART delivers competitive performance (91.4% on SST-2) with approximately 440K parameters—compared to 93.23% ([6]) using full BERT finetuning.

II. MATHEMATICAL PRINCIPLES

DART is inspired by a geometric and algebraic interpretation of dynamic model fusion. Consider two functions f_1 and f_2, each represented by a distinct geometric form, such as two lines in $\mathbb{R}^2$:

$$f_1(x,y) = a_1x + b_1y + c_1 = 0,$$
$$f_2(x,y) = a_2x + b_2y + c_2 = 0.$$

We define the ratio $\frac{f_1}{f_2} = \frac{0}{0}$, which is undefined and hence interpreted as an imaginary scalar controller η. This motivates a form where one function is expressed in terms of the other:

$$f_1 = \eta f_2 \Rightarrow a_1x + b_1y + c_1 = \eta(a_2x + b_2y + c_2).$$

Rearranging terms leads to a new equation:

$$(a_1 - a_2\eta)x + (b_1 - b_2\eta)y + (c_1 - c_2\eta) = 0$$

which defines a new straight line f_3. This line can dynamically interpolate between f_1 and f_2 depending on the value of η:

- If $\eta = 0$, $f_3 \equiv f_1$
- If $\eta \to \infty$, $f_3 \equiv f_2$
- If $0 < \eta < \infty$, f_3 lies between f_1 and f_2

However, η is unbounded, making it hard to optimize in practice. To regularize it, we introduce:

$$\lambda = \frac{1}{1+\eta}, \quad \text{so that } \lambda \in (0,1).$$

Now, we reformulate the fusion equation using a convex combination:

$$f_4 = \lambda f_1 + (1-\lambda)f_2 = 0.$$

This produces a new generic equation f_4 which smoothly interpolates between f_1 and f_2:

- $\lambda = 1 \Rightarrow f_4 = f_1$
- $\lambda = 0 \Rightarrow f_4 = f_2$
- $\lambda \in (0,1) \Rightarrow f_4$ represents a family of new functions between f_1 and f_2

This fusion principle is not limited to straight lines. Let f_1 be a straight line and f_2 be a circle. The formulation $f_\lambda = \lambda f_1 + (1-\lambda)f_2$ now spans conic sections, offering a continuous space of interpolated geometry. We generalize further: if f_1 and f_2 are two non-linear functions approximated by neural networks, then:

$$f(x) = \lambda f_1(x) + (1-\lambda)f_2(x)$$

becomes a **dynamic neural fusion** model where λ governs the adaptive mixing ratio.

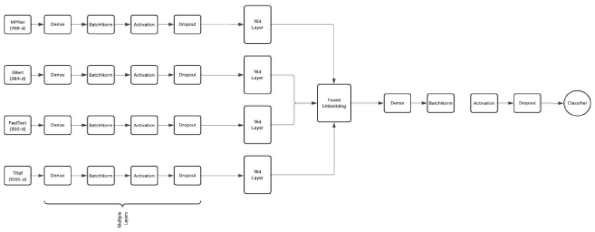
To enable per-dimension control, we extend λ from scalar to vector form $\boldsymbol{\lambda} \in [0,1]^d$. Let $\mathbf{v}_1$ and $\mathbf{v}_2$ be vector representations (embeddings) generated by two distinct networks or encoders, each projected into a shared space:

$$\tilde{\mathbf{v}} = \boldsymbol{\lambda}_1 \odot \mathbf{v}_1 + (1 - \boldsymbol{\lambda}_2) \odot \mathbf{v}_2,$$

where $\odot$ is element-wise multiplication. This fused embedding $\tilde{\mathbf{v}}$ is then passed to a downstream neural network for classification or regression.

In DART, $\boldsymbol{\lambda}$ is learned via a shallow controller network (tau), which dynamically produces the fusion weights for each instance. This construction allows DART to generalize over multiple representations, interpolate between learned functions, and remain interpretable while reducing training complexity and parameter overhead.

III. Architecture Overview



DART architecture: four diverse embeddings are projected, fused via per-input τ_i vectors, and passed to a lightweight classifier.

The DART (Dynamic Attention-based Reinforced Tau) architecture is designed to achieve strong performance in resource-constrained NLP settings by learning to dynamically fuse multiple pretrained embeddings with minimal parameter overhead. Figure 1 shows the overall structure of the system.

A. Input Representations

DART incorporates four heterogeneous embeddings, each offering a distinct view of the input sentence:

•MPNet (768D): captures contextual dependencies using masked and permuted language modelling.

•SBERT (384D): encodes semantic similarity through sentence-level contrastive training.

•FastText (300D): learns local and subword-level representations.

•TF-IDF (1000D): encodes term frequency-based statistical salience.

These embeddings are treated as non-trainable and serve as the base features for the fusion process. Their diversity reflects orthogonal linguistic and statistical properties.

B. Projection and Normalization

Each embedding is passed through a dedicated projection network to reduce its dimensionality and bring it into a shared latent space.

All projection networks share the same architecture:

- Dense layer (512 units) with L2 regularization,
- Batch Normalization to stabilize learning,
- LeakyReLU activation for non-linearity,
- Dropout for regularization,
- Final Dense layer projecting to a 16D latent space.

This results in four projected embeddings $\tilde{f}_i \in \mathbb{R}^{16}$, which act as the primary feature candidates for fusion.

C. Fusion Controller: τ Network

The centrepiece of the DART architecture is the τ fusion controller, inspired by dynamic gating and attention mechanisms. Instead of fixed or static weights, DART learns per-sample fusion weights by analyzing the latent embeddings jointly.

D. Embedding Disagreement Modeling via Absolute Differences

A distinguishing feature of the DART architecture is its use of a custom disagreement-aware fusion layer—Abs—designed to capture the mutual dissimilarity between projected embeddings. Traditional fusion strategies such as concatenation or averaging implicitly assume that all embeddings contribute equally or complementarily. However, this assumption is often invalid, especially when embeddings encode semantically orthogonal or even conflicting information.

To mitigate this, we compute the element-wise absolute differences between each pair of the four projected embeddings: MPNet (f_1), FastText (f_2), SBERT (f_3), and TF-IDF (f_4). The operation is defined as:

$$\text{Abs}(f_1, f_2, f_3, f_4) = \sum_{i<j} |f_i - f_j|$$

This summation captures the magnitude of disagreement across all six unique embedding pairs:

$$\{|f_1 - f_2|, |f_1 - f_3|, |f_1 - f_4|, |f_2 - f_3|, |f_2 - f_4|, |f_3 - f_4|\}$$

By design, this layer acts as a signal amplifier for embeddings that diverge significantly in their representation of the same input. For example, contextual embeddings like MPNet may emphasize long-range dependencies, while statistical embeddings like TF-IDF rely on term frequency. The disagreement between such representations, especially for nuanced sentiment inputs, holds rich information not captured in the individual vectors themselves.

The output of the `Abs` layer is then concatenated with the original projected embeddings to form the fusion input:

$$\text{fusion} = [f_1, f_2, f_3, f_4, \text{Abs}(f_1, f_2, f_3, f_4)]$$

This augmented feature vector serves two purposes:

1. It acts as the input to the τ controller sub-networks, guiding them with both raw semantic content and pairwise divergence cues.
2. It enables the model to focus on embeddings that are not only individually strong but also collectively diverse, thereby improving the expressivity of the fused representation.

This approach is grounded in the hypothesis that conflicting views can be as informative as consistent ones, particularly in tasks involving ambiguity, subjectivity, or domain shift. By explicitly encoding this disagreement signal, the DART framework enables more nuanced, context-aware fusion.

Moreover, this fusion mechanism avoids the pitfalls of over-reliance on dominant embeddings by incentivizing diversity and contrast, leading to improved generalization and robustness across samples with heterogeneous linguistic patterns.

$$\text{concatenate} = [\tilde{f}_1, \tilde{f}_2, \tilde{f}_3, \tilde{f}_4, |\tilde{f}_i - \tilde{f}_j|]$$

This fused context vector is passed through four separate sub-networks, each producing a 16D tau vector τ_i, one for each embedding:

$$\tau_i = \text{MLP}_i(\text{concatenate})$$

Each τ_i is activated via sigmoid to constrain its values between 0 and 1, functioning as a per-dimension reweighting vector.

E. Dynamic Fusion

The reweighted embeddings are then aggregated as:

$$\text{fused} = \sum_{i=1}^{4} \tau_i \odot \tilde{f}_i$$

where $\odot$ denotes element-wise multiplication.

This dynamic fusion mechanism is conceptually similar to attention but more interpretable and parameter-efficient. Instead of computing query-key-value attention scores, τ_i functions as a controllable gate for each embedding dimension.

F. Classification Head

The fused vector is passed through a compact classifier network comprising:

Dense(512) → BatchNormalization → LeakyReLU → Dropout(0.25)

Dense(128) → BatchNormalization → LeakyReLU → Dropout(0.25)

Dense(16) → BatchNormalization → LeakyReLU → Dropout(0.3)

Final Dense(2) → Softmax

Label smoothing is applied to the loss function to improve generalization, and Adagrad is used as the optimizer to adaptively scale learning rates across parameters.

G. Parameter Efficiency

Despite involving four different embeddings, DART maintains a lightweight footprint of approximately 440K trainable parameters. This is more than 200× smaller than a typical BERT-base finetuning setup (110M), yet achieves competitive results. This makes DART highly deployable in low-resource environments without sacrificing performance.

H. Training Strategy and Optimization Design

To complement the architectural efficiency of DART, we adopt a multi-phase training strategy rooted in the concept of warm starts and adaptive optimization. This approach ensures that the dynamic fusion controller τ and the classifier converges robustly with minimal overfitting, even in low-resource settings.

I. Warm Start with Progressive Optimizers

We begin training the network using the Adamax optimizer for a few epochs. Adamax, a variant of Adam based on the infinity norm, provides robust updates in the initial phase, especially in high-dimensional sparse settings such as TF-IDF inputs. Its ability to quickly stabilize learning dynamics helps initialize the fusion controller τ and the classifier weights toward a reasonable region in parameter space.

After the initial warm-up phase, we switch to RMSprop for fine-grained gradient control. RMSprop adapts the learning rate for each parameter using a moving average of squared gradients. This facilitates more refined convergence and avoids overshooting in sharp minima, which is particularly beneficial for networks with multiple small dense layers like those in DART.

Finally, the model is fine-tuned using Adagrad, which accumulates the square of gradients and scales each parameter's learning rate inversely. Adagrad is especially effective in sparse environments and has been shown to generalize better in NLP classification tasks. This staged transition helps combine the aggressive early learning of Adamax, the stabilizing nature of RMSprop, and the generalization strength of Adagrad.

J. Loss Function and Regularization

We employ the Binary Cross-Entropy loss with label smoothing (ϵ=0.1), which helps prevent the model from becoming overconfident in its predictions and improves generalization. This is critical in sentiment classification where language ambiguity can create semantically borderline inputs.

Dropout is applied at multiple points within both the projection networks and the classification head. Dropout rates range from 0.2 to 0.3, striking a balance between reducing co-adaptation and preserving learning capacity.

K. Callbacks and Training Control

To avoid overfitting and optimize training efficiency, we incorporate three essential callbacks:

- **ReduceLROnPlateau**: Monitors validation loss and reduces learning rate by a factor of 0.1 if performance stagnates, with a floor of $1e^{-7}$.
- This allows the model to escape potential plateaus in the loss landscape.
- **EarlyStopping**: Monitors validation loss and halts training if it fails to improve for 15 consecutive epochs. This prevents wasteful computation and mitigates overfitting risks.
- **ModelCheckpoint**: Saves the best model based on validation accuracy. Only the model with the highest generalization performance is retained for final evaluation.

This orchestration of callbacks enables dynamic learning rate adaptation, early convergence, and robust model selection with minimal manual intervention.

L. Motivation for Modular Design

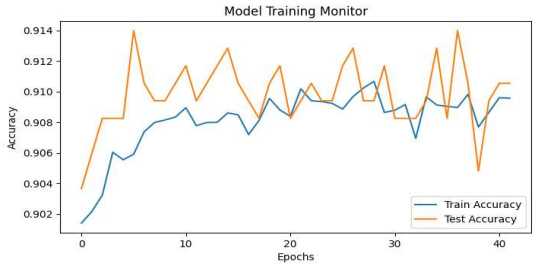
The overall architecture is modular by design, with independent tau-generating sub-networks for each embedding. This modularity ensures that the system remains interpretable and extensible—new embeddings can be added with minimal re-engineering. The separation between embedding projection, fusion weighting, and classification allows for better debugging, inspection, and incremental enhancements.

Moreover, the relatively shallow depth of the classifier and tau networks ensures that the full model remains lightweight (approx. 440K parameters), which is more than 200× smaller than a BERT-base finetuned model (110M). Despite this, DART achieves competitive accuracy (91.4%) on SST-2, showing that smart fusion, dynamic control, and optimization can outperform brute-force finetuning in constrained regimes.

IV. Benchmarking and Validation Results

To evaluate the efficacy of the DART architecture, we benchmarked its performance on the Stanford Sentiment Treebank (SST-2) dataset. Our primary metric is validation accuracy.

Figure 2 presents the validation accuracy over training epochs. The model demonstrates consistent improvement with early stopping triggered after approximately 18 epochs, achieving a peak accuracy of 91.4%.



Validation Accuracy across Epochs for DART on SST-2

A. Comparison with Baselines

We compare DART against several baselines in Table 1. DART achieves near-competitive performance to full BERT finetuning, while using <0.5% of the trainable parameters.

Validation Accuracy Comparison on SST-2

Model	*Accuracy %*	*Training Params*
TF-IDF + Logistic Regression	85.2	~10k
FastText + Dense Layer	87.0	~50k
BERT-base (finetuned)	93.4	~110M
DART (Ours)	91.4	~440k

V. Future Work : Reinforcement Learning Principles for Tau Optimization

While the initial version of the DART architecture learns the fusion controller τ using supervised learning and backpropagation, such a strategy may not generalize well to complex, non-linear relationships among embeddings. To overcome this, we propose augmenting τ with reinforcement learning (RL), enabling instance-specific, reward-driven optimization.

A. Limitations of Supervised Optimization

In standard supervised learning, τ is learned by minimizing a differentiable loss function such as cross-entropy. However, this approach:

- Provides no exploration of alternative fusion strategies.
- Propagates gradients only from output layers, missing latent interactions.
- May converge to sub-optimal fusion due to local minima.

To resolve this, we treat the selection of τ as a decision-making process guided by a learned policy.

B. Policy-Based Formulation

We define a policy $\pi_\theta(\tau \mid x)$, parameterized by θ, that outputs a distribution over possible τ values conditioned on input x. The objective is to maximize the expected reward:

$$\mathcal{J}(\theta) = \mathbb{E}_{\tau \sim \pi_\theta(\cdot|x)}[R(\tau, x)],$$

where $R(\tau, x)$ is a scalar reward (e.g., classification accuracy or validation F1).

C. Policy Gradient Optimization

The policy is updated using the REINFORCE algorithm:

$$\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[\nabla_\theta \log \pi_\theta(\tau|x)(R(\tau) - b)],$$

where b is a baseline to reduce variance in gradient estimates.

D. Stochastic Exploration

Unlike deterministic fusion, this formulation allows stochastic exploration:

$$\tau \sim \pi_\theta(\cdot\,|x)$$

which enables the model to sample diverse τ vectors and adapt based on reward feedback. This is particularly valuable in DART, where different inputs benefit from different fusion strategies.

E. Credit Assignment

Rewards serve as feedback to assign credit to beneficial τ values. For instance, if $\tau = [0.7, 0.2, 0.05, 0.05]$ yields improved performance, the policy is updated to increase the likelihood of similar fusion vectors.

F. Generalization Benefit

By treating τ as a policy instead of a fixed parameter vector, we allow it to:

- Learn context-aware fusion for each input.
- Adapt dynamically over time based on performance.
- Generalize better across diverse input distributions.

Thus, reinforcement learning serves as a principled mechanism for discovering effective and personalized fusion strategies in the DART framework.

G. Incorporating Attention over Fused Representations

While reinforcement learning provides a global control mechanism for guiding τ, attention offers a complementary approach that enables localized, fine-grained control over the fused representation space. We propose extending DART \ applying multi-head self-attention across the fused embeddings before classification.

H. Attention Motivation

Each projected embedding $\tilde{f}_i$ (after scaling by τ_i) captures different facets of the input: semantic, syntactic, local, or statistical. Their naive summation assumes all elements of each vector contribute equally, which may not hold. Attention enables DART to selectively focus on the most relevant dimensions across all fused embeddings.

I. Formulation

Let the dynamically weighted embeddings be:

$$F = [\tau_1 \cdot \tilde{f}_1,\ \tau_2 \cdot \tilde{f}_2,\ \ldots,\ \tau_n \cdot \tilde{f}_n] \in \mathbb{R}^{n\times d},$$

where n is the number of input sources and d is the projected dimensionality (e.g., 16).We compute multi-head attention as:

$$Q = FW_Q, \quad K = FW_K, \quad V = FW_V,$$

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V.$$

The result is a context-enhanced embedding that learns interdependence between embeddings:

$$F_{\text{attn}} = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W_O,$$

where each attention head models different relational patterns among the fused embeddings.

J. Benefits

- Attention enhances feature interaction across embeddings.
- Reduces redundancy in fused vectors by weighting informative channels.
- Supports multi-modal or multi-lingual fusion through flexible scaling.
- Encourages interpretability by exposing which features dominate decision-making.

K. Integration with Tau

Attention is applied after dynamic weighting with τ, making it a downstream mechanism that sharpens and redistributes the fused features based on inter-feature correlation. This layered structure—controller followed by attention—mirrors gating in memory networks and allows DART to act both as a policy learner and attention aggregator.

L. Reinforcement-Driven Attention Optimization

Additionally, the output of attention layers can be treated as part of the reward function in reinforcement learning. This hybrid architecture allows the model to co-optimize τ and attention jointly for better downstream accuracy and robustness across varied tasks.

These results reinforce our core hypothesis: *smart embedding fusion with lightweight adaptive controllers can achieve high accuracy with dramatically fewer parameters and training resources.*

VI. Comparison with Gating Mechanisms, MoE, and Attention-Based Fusion

The fusion strategy employed in DART is related to several established paradigms in the literature, including gating networks, Mixture-of-Experts (MoE) (**[5]**), and attention-based fusion (**[2]**). In this section, we outline the similarities and key differences to clarify the novel aspects of our approach.

A. Relation to Gating Mechanisms

Gating mechanisms typically learn a scalar or vector gate $g \in [0,1]^n$ for modulating multiple input paths:

$$f_{\text{gated}}(x) = \sum_i g_i(x) \cdot f_i(x),$$

where $f_i(x)$ denotes different inputs or expert outputs. These gates are often learned jointly with the main task, using shallow neural layers or parameterized functions of the input.

B. Difference from DART:

- In DART, fusion is not performed over intermediate network branches but over pretrained, fixed embeddings that represent diverse linguistic information.
- Instead of learning gates based solely on input, DART includes a disagreement signal—computed as the sum of absolute pairwise differences between projected embeddings—which provides a sense of semantic variance across inputs.
- The τ-based fusion controller operates after projecting all embeddings into a shared low-dimensional space, which is a critical design choice to reduce parameter complexity and improve compatibility across heterogeneous representations.

C. Relation to Mixture-of-Experts (MoE)

MoE architectures typically consist of multiple full networks (experts) with a learnable gating function ([5]) that assigns routing weights:

$$f_{\text{moe}}(x) = \sum_i \alpha_i(x) \cdot \text{Expert}_i(x).$$

While MoEs are effective in scaling large models, they often require auxiliary losses for balancing expert usage and maintaining stability during training.

D. Difference from DART:

- DART does not route inputs through full neural networks but instead uses precomputed, pretrained embeddings (e.g., MPNet, SBERT) as static inputs, thus reducing training time and compute cost.
- The fusion weights in DART are learned via a lightweight MLP and trained end-to-end with the downstream task, without any need for auxiliary load-balancing objectives.
- The model size of DART remains fixed and compact, avoiding the scaling pitfalls of typical MoE models.

E. Relation to Attention-Based Fusion

Attention mechanisms, especially in multimodal and multi-representation settings, compute dynamic relevance scores for each input channel based on content similarity:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V.$$

This is often used to selectively aggregate information across sources or modalities.

F. Difference from DART:

- DART's fusion does not require the explicit construction of query-key matrices, making it more computationally efficient.
- While attention models learn pairwise interactions explicitly, DART uses the disagreement vector and τ network to learn fusion weights implicitly from the data.
- Attention could be added as a complementary mechanism on top of DART's fused embeddings, which we highlight as a direction for future work.

G. Discussion on Uniqueness and Limitations

The core strength of DART lies in its simplicity and grounding in mathematically interpretable constructs (Section III). By treating fusion as a learned convex combination of projected embeddings, DART avoids the complexity of MoE architectures while being more adaptive than static gating.

However, DART assumes that the input embeddings are already semantically rich and complementary, and does not perform joint finetuning of the embeddings themselves. This could limit its applicability when upstream embeddings are poorly aligned. Additionally, the τ controller, though efficient, might underperform in scenarios requiring high-capacity reasoning unless further scaled or guided using auxiliary signals (e.g., reinforcement feedback or attention).

REFERENCES

[1] E. Strubell, A. Ganesh, and A. McCallum, Energy and policy considerations for deep learning in NLP," in *Proc. ACL*, 2019.

[2] A. Vaswani et al., Attention is all you need," in *NeurIPS*, 2017.

[3] J. Devlin et al., BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019.

[4] T. Brown et al., Language models are few-shot learners," in *NeurIPS*, 2020.

[5] N. Shazeer et al., Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," in *ICLR*, 2017.

[6] J. Zang, "bert-base-uncased-sst2," Hugging Face, 2024. [Online]. Available: https://huggingface.co/JeremiahZ/bert-base-uncased-sst2