Element-wise Multiplicative Interactions in Neural Networks: Theory, Advances, and Open Problems

Gurpreet Inayatullah[1], Purnima Shafiq[1]

[1]Indian Institute of Technology Bombay, India
gurpreet.inayatullah@iitb.ac.in , purnima.shafiq@iitb.ac.in

Abstract. The Hadamard product, or element-wise multiplication, has emerged as a fundamental operation in modern deep learning architectures, far beyond its origins in linear algebra. This survey provides a comprehensive overview of the Hadamard product's role across neural network models, with a particular focus on its use in gating mechanisms, attention modules, feature fusion strategies, and meta-learning systems. We begin by introducing the mathematical formulation and basic properties of the Hadamard product, and then explore its integration into deep learning through numerous architectural motifs. Recent advances reveal creative and powerful extensions of the operation, including learnable modulations, bilinear and trilinear interactions, stochastic variants, and structured sparsity mechanisms. Despite its advantages in computational efficiency and model flexibility, the Hadamard product also presents critical challenges, such as expressive limitations, dimensional rigidity, optimization instability, and a lack of theoretical grounding. This survey not only highlights these challenges but also outlines promising future research directions aimed at enhancing the utility, interpretability, and theoretical understanding of element-wise multiplicative interactions in deep learning. Our goal is to establish the Hadamard product as a first-class citizen in the design and analysis of next-generation neural architectures.

Keywords: Hadamard product, element-wise multiplication, deep learning, neural networks, gating mechanisms, attention, feature fusion, bilinear models, optimization, model interpretability.

1 Introduction

The field of deep learning has undergone tremendous growth over the past decade, leading to groundbreaking advances in diverse areas such as computer vision, natural language processing, reinforcement learning, and speech recognition. Central to many of these advances are fundamental mathematical operations that enable neural networks to efficiently model complex, high-dimensional data. Among these operations, the Hadamard product—also known as the element-wise or Schur product—has emerged as a critical yet often underappreciated

component in the design and optimization of modern deep learning architectures. The Hadamard product, denoted typically by the symbol $\odot$, is the element-wise multiplication of two matrices (or vectors) of identical dimensions [1]. Given two matrices A and B of the same size, their Hadamard product $C = A \odot B$ is a matrix of the same size where each entry $c_{ij} = a_{ij}b_{ij}$ [2]. Unlike the standard matrix product, the Hadamard product operates independently on corresponding elements, resulting in a straightforward but powerful form of interaction between two data structures. This simple operation has profound implications for model expressivity, computational efficiency, and the ability to incorporate various forms of inductive bias into neural network architectures. Historically, the Hadamard product was studied primarily in the context of linear algebra and matrix analysis, with applications in statistics, signal processing, and combinatorics [3]. However, its utility within deep learning frameworks has become increasingly evident as researchers seek methods to control information flow, model multiplicative interactions, and enhance feature representations. The advent of attention mechanisms, gating structures such as those found in LSTM (Long Short-Term Memory) networks, and bilinear models has highlighted the importance of element-wise interactions in capturing fine-grained relational patterns that might be obscured by purely additive or convolutional approaches [4]. One of the key reasons the Hadamard product is so widely adopted in deep learning is its compatibility with gradient-based optimization techniques. The operation is differentiable, computationally inexpensive, and amenable to parallelization, making it highly suitable for large-scale learning tasks [5]. Moreover, it introduces a form of nonlinearity and interaction that can often improve model capacity without substantially increasing computational burden. As models grow deeper and more complex, leveraging efficient operations like the Hadamard product becomes increasingly crucial in balancing expressivity with scalability [6]. The use of the Hadamard product extends beyond simple multiplicative interactions. In attention mechanisms, particularly the popular Transformer architecture, element-wise products are used to blend query, key, and value representations, modulating the importance of different input features dynamically [7]. Similarly, gating mechanisms in recurrent neural networks and various types of memory-augmented models rely on Hadamard products to selectively update hidden states, allowing the network to learn when and how much to forget or retain information over time [8]. In multimodal learning, where data from different modalities such as text, image, and audio must be fused, element-wise multiplication provides a simple yet effective means of capturing cross-modal interactions. Despite its wide adoption, the use of the Hadamard product is not without challenges [9]. Questions arise regarding its expressiveness compared to more complex interaction mechanisms, the risk of vanishing gradients in very deep networks when multiplicative operations are stacked, and the difficulty of interpreting models that rely heavily on element-wise interactions [10]. Furthermore, as deep learning applications expand into domains requiring greater robustness, interpretability, and fairness, understanding the theoretical properties and practical implications of operations like the

Hadamard product becomes increasingly important. In this survey, we systematically explore the role of the Hadamard product in deep learning, examining its mathematical foundations, practical applications, and recent innovations that leverage element-wise operations for improved performance and generalization. We begin by reviewing the basic properties and theoretical considerations of the Hadamard product in Section 2, followed by a detailed discussion of its applications across a variety of deep learning paradigms in Section 3. In Section 4, we highlight recent methodological advances that push the boundaries of traditional uses of the Hadamard product. Finally, in Section 5, we discuss open challenges, limitations, and potential future directions for research involving element-wise operations in deep learning. Through this comprehensive examination, we aim to provide researchers and practitioners with a deeper understanding of how a seemingly simple mathematical operation plays a pivotal role in shaping the capabilities and evolution of modern deep learning systems. By shedding light on the nuances of the Hadamard product, we hope to inspire further exploration into its theoretical underpinnings, novel applications, and optimization within increasingly sophisticated models [11].

2 Mathematical Background

To fully appreciate the role of the Hadamard product in deep learning, it is essential to understand its mathematical properties, its relationship with other matrix operations, and its behavior under various transformations [12]. This section provides a comprehensive overview of the Hadamard product from a linear algebraic perspective, establishing the theoretical foundation necessary for its application in neural network design.

2.1 Definition and Basic Properties

Let $A = [a_{ij}]$ and $B = [b_{ij}]$ be two matrices of the same dimension $m \times n$, where $a_{ij}, b_{ij} \in \mathbb{R}$. The Hadamard product of A and B, denoted $A \odot B$, is defined as:

$$(A \odot B)_{ij} = a_{ij} b_{ij}, \quad \text{for all } i = 1, \ldots, m, \quad j = 1, \ldots, n[13].$$

This operation is performed element-wise, resulting in a new matrix $C = A \odot B \in \mathbb{R}^{m \times n}$ [14]. Unlike the standard matrix product, the Hadamard product does not involve summation over indices and is therefore commutative:

$$A \odot B = B \odot A.$$

It is also associative and distributive with respect to matrix addition:

$$(A \odot B) \odot C = A \odot (B \odot C), \quad A \odot (B + C) = A \odot B + A \odot C.$$

2.2 Comparison with Other Matrix Products

The Hadamard product should not be confused with the standard (dot) matrix product or the Kronecker product. While the matrix product AB involves summation across rows and columns and reflects linear transformations between vector spaces, the Hadamard product lacks such geometric interpretation [15]. Nonetheless, it is powerful in contexts where direct, localized interactions between individual components are desired. In contrast to the Kronecker product, which produces a large block matrix capturing tensor-like behavior, the Hadamard product is computationally simpler and memory-efficient [16]. These properties make it well-suited for deep learning, where matrix dimensions can be large and real-time inference efficiency is crucial [17].

2.3 Relation to Element-wise Nonlinearities

In deep learning, element-wise operations are ubiquitous [18]. The Hadamard product can be viewed as a multiplicative counterpart to element-wise nonlinear activations such as $\mathrm{ReLU}(x)$ or $\tanh(x)$, and often appears in conjunction with them [19]. For example, many gating mechanisms in neural networks involve applying a sigmoid activation to one input and then performing a Hadamard product with another input:

$$h = \sigma(Wx + b) \odot \tanh(Vx + c),$$

where σ denotes the sigmoid function and tanh denotes the hyperbolic tangent [20]. This formulation allows the model to dynamically modulate the flow of information, a crucial capability for tasks involving temporal sequences or hierarchical representations [**?**].

2.4 Spectral Properties

An interesting property of the Hadamard product is its effect on the spectral (eigenvalue) characteristics of matrices [21]. While the eigenvalues of the standard matrix product are well-understood in terms of linear transformations, the eigenvalues of the Hadamard product are not simply related to those of the operand matrices [22]. However, several useful inequalities and identities are known [23]. For example, the Schur product theorem states that if A and B are positive semidefinite (PSD) matrices, then $A \odot B$ is also PSD:

$$A \succeq 0, \quad B \succeq 0 \quad \Rightarrow \quad A \odot B \succeq 0.$$

This property is particularly important in probabilistic modeling and kernel methods, where preserving positive semidefiniteness is a key constraint.

2.5 Hadamard Product and Differentiability

In the context of deep learning, any operation used within a neural network must be differentiable to enable gradient-based optimization. The Hadamard product satisfies this requirement. Given differentiable functions $f(x)$ and $g(x)$ with vector outputs, the gradient of their Hadamard product $h(x) = f(x) \odot g(x)$ with respect to x is given by:

$$\nabla h(x) = \nabla f(x) \odot g(x) + f(x) \odot \nabla g(x),$$

which follows directly from the product rule for derivatives applied element-wise [24, 25]. This makes the Hadamard product particularly appealing for constructing differentiable computational graphs.

2.6 Computational Considerations

From a computational standpoint, the Hadamard product is extremely efficient [26]. It requires only $O(n)$ operations for vectors of length n (or $O(mn)$ for matrices of size $m \times n$), and can be implemented using highly optimized element-wise routines on CPUs and GPUs [27]. Its memory access patterns are also simple and regular, which helps maximize throughput on modern parallel hardware [28].

2.7 Interpretability and Visualization

Although the Hadamard product is easy to compute, its effect on model behavior can be difficult to interpret, particularly when used in high-dimensional, nonlinear systems [29]. Recent research has begun to explore methods for visualizing and understanding the impact of multiplicative interactions on learned representations [30]. In some cases, Hadamard products have been shown to act as soft feature selectors or modulators, amplifying or suppressing features based on context [31].

2.8 Theoretical Extensions and Generalizations

Recent theoretical work has proposed generalizations of the Hadamard product, such as the p-Hadamard product, which introduces a power-law scaling between element-wise inputs, or probabilistic variants where element-wise multiplications are replaced by expected interactions under noise distributions [32]. These extensions provide a richer framework for modeling uncertainty and structural dependencies, and open new avenues for incorporating domain-specific priors into learning systems [33]. Overall, the Hadamard product combines elegant mathematical properties with practical utility, making it a valuable tool in both the theoretical and applied domains of deep learning. In the following section, we examine how this operation is concretely employed across a variety of neural architectures and learning paradigms [34].

3 Applications of the Hadamard Product in Deep Learning

The Hadamard product plays a foundational role in numerous deep learning architectures and paradigms, often serving as a mechanism for interaction, gating, modulation, or fusion [35]. Despite its simple formulation, its inclusion enables models to express complex behaviors through local multiplicative interactions, which can be more effective or efficient than additive operations in certain contexts [36]. This section surveys a broad range of applications where the Hadamard product has demonstrated significant utility.

3.1 Gating Mechanisms in Recurrent Neural Networks

One of the earliest and most impactful uses of the Hadamard product in deep learning is found in gating mechanisms of Recurrent Neural Networks (RNNs), particularly in Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs). These models introduce gates—such as input, forget, and output gates—that control the flow of information through the recurrent architecture [37]. Each gate computes a value via a sigmoid activation and subsequently modulates another vector through element-wise multiplication:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f), \quad c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t [38].$$

Here, the Hadamard product enables selective updating of the cell state c_t, allowing the network to learn long-term dependencies and mitigate the vanishing gradient problem that plagues traditional RNNs [39].

3.2 Attention Mechanisms and Transformers

In attention-based models, especially the Transformer architecture, the Hadamard product is used to modulate and combine information from multiple sources [40]. For example, in multi-head attention, after computing attention scores and applying the softmax function, attention weights are often used to scale value vectors via element-wise multiplication before summing:

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

While this formula uses matrix multiplication in its standard form, several extensions and simplifications, such as in Linformer or Efficient Attention variants, incorporate Hadamard products to reduce complexity and inject learned biases [41]. Additionally, cross-attention mechanisms in multimodal models often use the Hadamard product to fuse feature representations across modalities [42].

3.3 Feature Fusion in Multimodal Learning

Multimodal learning involves integrating information from different sensory modalities such as text, image, audio, and video. The Hadamard product provides an efficient and interpretable method of combining feature vectors from different modalities [43]. Suppose $x \in \mathbb{R}^d$ represents a visual feature and $y \in \mathbb{R}^d$ a textual embedding; their fused representation can be expressed as:

$$z = x \odot y[44].$$

This formulation preserves dimensionality and introduces multiplicative interactions between aligned dimensions, allowing for richer representations than simple concatenation or addition. Models such as Multimodal Compact Bilinear Pooling (MCB) and Low-rank Bilinear Pooling (MLB) extend this idea by applying Hadamard products followed by projection, demonstrating strong performance on tasks like Visual Question Answering (VQA) [45].

3.4 Element-wise Modulation in Conditional Computation

The Hadamard product also serves as an efficient mechanism for implementing conditional computation in neural networks, where parts of the network are selectively activated based on input-dependent gates or conditions. For instance, in dynamic routing, feature masks computed from an auxiliary network can modulate base network activations:

$$\hat{x} = m(x) \odot x,$$

where $m(x)$ is a learned gating function [46]. Such techniques improve model efficiency and interpretability by allowing selective activation and suppressing irrelevant pathways during inference.

3.5 Neural Tensor Networks and Bilinear Models

In certain relation modeling tasks, such as those found in knowledge graphs or question answering systems, bilinear interactions between vectors are crucial. While full tensor-based approaches are computationally expensive, simplified versions often rely on Hadamard products to capture pairwise interactions efficiently. For instance, a scoring function for a relation r between entities e_1 and e_2 might take the form:

$$f(e_1, r, e_2) = (W_r e_1)^\top (e_2 \odot v_r),$$

where v_r is a relation-specific embedding [47]. This use of the Hadamard product introduces multiplicative conditioning of the input on the relation, allowing for fine-grained semantic interactions with reduced parameterization.

3.6 Normalization and Regularization Techniques

In certain normalization schemes, the Hadamard product is employed to scale normalized features by learned parameters. For instance, Layer Normalization and Adaptive Instance Normalization (AdaIN) use scale-and-shift parameters γ and β, often applied element-wise:

$$\text{Norm}(x) = \gamma \odot \hat{x} + \beta[48].$$

Such schemes enable the model to adaptively control feature magnitudes and learn feature importance dynamically, which has proven critical in tasks involving style transfer, domain adaptation, and generative modeling [49].

3.7 Efficient Representation Learning

The Hadamard product is particularly advantageous in architectures designed for low-resource or real-time settings [50]. Its simplicity allows for compact neural modules that can be deployed on edge devices or mobile hardware. In binary neural networks and quantized models, element-wise multiplication can be approximated or even replaced with logical operations, offering further computational savings while retaining effective feature interaction modeling.

3.8 Meta-learning and Parameter Modulation

In meta-learning and few-shot learning scenarios, models must rapidly adapt to new tasks with limited data [51]. Hadamard products are often used to enable task-specific modulation of model parameters [52]. For example, in parameter-efficient fine-tuning techniques, task embeddings are used to produce scaling vectors that modulate the base model's parameters or activations:

$$\theta' = \theta \odot \phi(z),$$

where $\phi(z)$ is a task-conditioned modulator. This approach enables flexible transfer learning and improves generalization across diverse tasks [53].

3.9 Applications in Generative Models

Generative models, particularly those based on variational inference and adversarial training, also benefit from the use of Hadamard products. In Variational Autoencoders (VAEs), the reparameterization trick commonly involves an element-wise product between a learned standard deviation and a noise sample:

$$z = \mu + \sigma \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)[54].$$

This operation is crucial for enabling gradient flow through stochastic sampling processes. Similarly, in style-based GANs such as StyleGAN, Hadamard products are used for adaptive modulation of convolutional activations, providing fine control over the synthesis process [55].

3.10 Summary

The ubiquity of the Hadamard product in deep learning reflects its versatility, computational efficiency, and ability to represent complex relationships with minimal overhead [56]. Its applications span across nearly every subfield of deep learning—from sequence modeling and vision to multimodal reasoning and generative modeling [57]. In many cases, the Hadamard product offers a balance between expressivity and simplicity, enabling models to capture non-trivial interactions without incurring prohibitive computational costs. In the following section, we delve deeper into the innovations and advances that extend the standard Hadamard product, examining how researchers have adapted, generalized, and optimized it to meet the evolving demands of modern neural networks.

4 Recent Advances

While the Hadamard product has long been recognized as a basic mathematical operation, recent research in deep learning has revealed its deeper potential as a core mechanism for encoding structure, implementing adaptive control, and enabling efficient representation learning [58]. These innovations often go beyond the classical formulation of the Hadamard product, introducing generalized forms, learnable interaction schemes, or compositional models that build on the element-wise paradigm [59]. In this section, we explore a selection of significant methodological advances that extend or refine the role of the Hadamard product in contemporary neural networks.

4.1 Learnable Element-wise Modulation

A major innovation in the use of the Hadamard product is the incorporation of learnable element-wise modulation, where the scaling vectors applied via $\odot$ are not fixed or input-dependent, but instead parameterized and trained alongside the model. Consider a representation $x \in \mathbb{R}^d$ and a learned modulation vector $m \in \mathbb{R}^d$. The modulated output is given by $y = m \odot x$. This form of gating introduces multiplicative parameters that selectively amplify or suppress individual dimensions of the feature space. Such mechanisms appear prominently in architectures like Squeeze-and-Excitation (SE) networks, where a global descriptor is extracted via average pooling and then passed through a small bottleneck network to generate channel-wise modulation weights:

$$s = \sigma(W_2\,\mathrm{ReLU}(W_1\,z)), \quad \text{where } z = \frac{1}{H \times W}\sum_{i=1}^{H}\sum_{j=1}^{W} x_{ij}[60].$$

The output $s \in \mathbb{R}^C$ is then broadcast and multiplied with the original feature map x via the Hadamard product, allowing dynamic recalibration of feature channels [61]. This technique has been generalized in attention modules, such as the CBAM (Convolutional Block Attention Module), and has proven effective in tasks requiring spatial and channel-wise sensitivity [62].

4.2 Hadamard Product in Hypernetworks and Meta-Learning

In the context of meta-learning and parameter prediction, the Hadamard product is used to condition a base network on task-specific embeddings [63]. Let θ denote the base parameters of a network and let $\phi(z)$ represent a task embedding derived from a meta-learner. A modulated parameterization may be written as:

$$\tilde{\theta} = \theta \odot \phi(z),$$

where $\phi(z)$ is typically a vector produced by a small neural network that encodes contextual information [64]. This approach, often referred to as *parameter modulation* or *feature-wise transformation*, enables the model to adapt its behavior without altering its structure. It is especially useful in few-shot learning, domain adaptation, and continual learning scenarios, where fast and efficient generalization is essential [65]. Some extensions incorporate affine modulation, adding a bias term: $\tilde{\theta} = \theta \odot \phi_1(z) + \phi_2(z)$, which has strong theoretical connections to feature-wise linear modulation (FiLM) layers [66]. These advances demonstrate that the Hadamard product is not just a computational shortcut but a flexible interface for conditioning and adaptation in modern learning systems.

4.3 Generalized Bilinear and Tri-linear Models

Several researchers have sought to generalize the Hadamard product to capture higher-order interactions between input representations. One direction involves the use of bilinear or trilinear pooling mechanisms, where multiple vectors are combined using element-wise multiplication followed by a projection:

$$f(x, y) = W^\top (x \odot y), \quad f(x, y, z) = W^\top (x \odot y \odot z),$$

with $x, y, z \in \mathbb{R}^d$ and $W \in \mathbb{R}^d$ or higher-order tensors in more expressive variants [67]. These models are particularly prominent in tasks like Visual Question Answering (VQA), where interactions between visual and linguistic modalities must be richly represented [68]. To address the dimensionality and parameter burden, several approximations have been proposed, such as compact bilinear pooling using Count Sketch projections, or low-rank factorization of the projection matrix. These methods retain the expressive power of the Hadamard interaction while remaining computationally tractable, making them suitable for deployment in real-time systems and on large-scale datasets.

4.4 Probabilistic and Stochastic Interpretations

Another frontier of research interprets the Hadamard product in probabilistic terms. In variational models, the reparameterization trick involves the Hadamard product to ensure differentiability through sampling [69]. Given a latent variable $z = \mu + \sigma \odot \epsilon$, with $\epsilon \sim \mathcal{N}(0, I)$, the element-wise product with the standard deviation σ enables learning of heteroscedastic uncertainty while maintaining

gradient flow [70]. Beyond this, some models incorporate stochastic gating or noise-injected modulation schemes where the gating vector g is a random variable:

$$x' = x \odot g, \quad g_i \sim \text{Bernoulli}(p_i) \text{ or } \mathcal{N}(1, \sigma_i^2).$$

Such stochastic variants serve as regularizers, akin to Dropout, and have been studied for their ability to improve generalization, promote sparsity, and increase robustness to noise [71].

4.5 Algebraic Extensions: *p*-Hadamard and Log-space Variants

Mathematically inspired extensions of the Hadamard product aim to generalize its behavior or apply it in alternative domains [72]. One such generalization is the p-Hadamard product, defined as:

$$(A \odot_p B)_{ij} = (a_{ij}^p \cdot b_{ij}^p)^{1/p},$$

which recovers the standard Hadamard product when $p = 1$ but allows for more flexible forms of multiplicative interaction when $p \neq 1$ [73]. This operation maintains commutativity and provides a tunable way to interpolate between linear and multiplicative effects. Another direction is the use of element-wise operations in log-space, where multiplication becomes addition. That is, for positive inputs $x, y > 0$, one may compute:

$$\log(x \odot y) = \log x + \log y.$$

This formulation allows certain optimization techniques to operate in additive domains, simplifying gradients or allowing direct application of convex optimization methods in models where positivity and scale are critical constraints.

4.6 Structured Sparsity and Interpretability

Finally, recent advances have leveraged the Hadamard product as a means to induce structured sparsity in neural networks [74]. By applying learned masks or attention weights element-wise, models can identify and prune irrelevant or redundant dimensions dynamically. For example, a gating vector $g = \sigma(Wx)$ applied as $x' = g \odot x$ allows only a subset of features to propagate forward, with the rest effectively suppressed. When g is sparse, this leads to efficient subnetwork selection, interpretability, and potential hardware acceleration through model compression [75]. Moreover, the Hadamard product's localized behavior has been used in saliency mapping and feature attribution. By observing the sensitivity of outputs to perturbations in the gating vector or its gradient, researchers can gain insight into which input components are most critical for a given prediction [76].

4.7 Summary

These recent advances underscore the Hadamard product's evolution from a basic mathematical tool to a fundamental building block in deep learning innovation. Its flexibility, differentiability, and compatibility with neural architectures allow it to serve as both a computational primitive and a high-level design pattern [77]. Whether used to implement gates, fuse features, encode conditional structure, or approximate complex interactions, the Hadamard product continues to inspire new forms of model expressivity and efficiency [78]. In the next section, we address the remaining challenges and open questions surrounding the use of the Hadamard product in deep learning, including theoretical concerns, practical limitations, and emerging directions for future research.

5 Challenges and Open Problems

Despite its ubiquity and computational efficiency, the Hadamard product also presents several theoretical and practical challenges when employed in deep learning models. These challenges are not only rooted in the operation's mathematical limitations, but also arise from its interaction with the statistical, optimization, and generalization properties of neural architectures [79]. In this section, we discuss open problems and unresolved issues associated with the Hadamard product in modern learning systems [80].

5.1 Expressive Limitations of Element-wise Multiplication

One of the most fundamental concerns with the Hadamard product is its inherently local and dimension-wise nature [81]. For vectors $x, y \in \mathbb{R}^d$, the Hadamard product $x \odot y$ computes each output component as $z_i = x_i y_i$, without any cross-dimensional interaction [82]. This strict alignment constraint limits the expressive capacity of such interactions, particularly when richer, higher-order relationships between input features are required [83]. Unlike bilinear models or full matrix products, which can capture arbitrary linear transformations and dependencies between input features, the Hadamard product assumes strict independence across dimensions. This can be a bottleneck in tasks requiring reasoning about correlations, spatial structure, or contextual dependencies. Addressing this limitation often involves combining the Hadamard product with additional projection or attention mechanisms, but a formal characterization of its expressive power relative to other interaction schemes remains an open research question [84].

5.2 Gradient Flow and Optimization Instability

Although differentiable, the Hadamard product can contribute to instability in gradient propagation under certain conditions [**?**]. In particular, when the multiplicands involve sigmoidal activations or soft gates, such as in LSTMs or attention masks, the gradients through $x \odot \sigma(y)$ may vanish or explode depending on

the input scale. This problem is exacerbated when x and $\sigma(y)$ both tend toward zero, leading to minimal gradient signals and impeding learning [85]. Additionally, because the Hadamard product does not involve parameter sharing or reuse across dimensions, it can amplify noise and variation in feature vectors, particularly in early training stages. Careful initialization and normalization are often required to mitigate these effects, but no universal solution exists [86]. Recent work has proposed the use of gated residual connections and gradient clipping to improve stability, yet the precise role of the Hadamard product in gradient dynamics remains under-explored.

5.3 Over-reliance on Alignment and Dimensionality Matching

Another major limitation of the Hadamard product is its strict requirement for dimensional alignment. Since $x \odot y$ is only defined when $x, y \in \mathbb{R}^d$ for the same d, it imposes strong constraints on the network design, particularly in multimodal or heterogeneous feature fusion settings. When modalities or intermediate representations differ in size or structure, the use of Hadamard products necessitates auxiliary transformations such as padding, projection, or masking to enforce shape compatibility:

$$z = \phi(x) \odot \psi(y), \quad \text{with } \phi, \psi : \mathbb{R}^{d'} \to \mathbb{R}^d [87].$$

These transformations can introduce additional parameters, increase model complexity, or distort the original information, undermining the advantages of simplicity and efficiency. Moreover, aligning dimensions through learned projections may dilute modality-specific signals, particularly when applied aggressively in deep architectures [88].

5.4 Interpretability and Sparsity Control

While the Hadamard product enables intuitive feature gating and scaling, its interpretability is not guaranteed in practice [89]. The multiplicative interaction may suppress certain dimensions arbitrarily, depending on the optimization trajectory, without offering insight into why specific features are emphasized or attenuated. This poses challenges in domains requiring transparency and accountability, such as healthcare, finance, or law. Furthermore, although the Hadamard product is often associated with inducing sparsity (e.g., through element-wise masking), it does not inherently enforce structured or group sparsity without additional constraints. Researchers have proposed combining it with L_1 or group-lasso regularization to promote interpretable representations, but these techniques require careful tuning and validation [**?**]. A general framework for learning interpretable, structured masks via Hadamard modulation remains an important open problem [90].

5.5 Computational Trade-offs in Large-scale Architectures

Despite being computationally cheap in isolation, the Hadamard product may incur hidden costs in large-scale neural architectures when used extensively [91]. In particular, when element-wise operations are repeatedly applied in multi-branch networks, each multiplication may require memory allocation, synchronization, and bandwidth, especially in GPU and distributed environments [92]. Unlike matrix multiplications, which are highly optimized and batch-friendly, the Hadamard product often introduces irregular memory access patterns, leading to suboptimal hardware utilization [93]. This issue becomes pronounced in scenarios involving long sequences, high-resolution feature maps, or multiple interacting modalities, where the cumulative cost of many element-wise operations can become a bottleneck. Optimizing memory layouts, using fused kernels, or designing efficient dataflows are potential solutions, but they require low-level engineering effort and may not generalize across frameworks and platforms [94, 95].

5.6 Lack of Theoretical Frameworks and Guarantees

Finally, the widespread empirical use of the Hadamard product in deep learning is not matched by an equally robust theoretical understanding [96]. While its algebraic properties—commutativity, associativity, and distributivity—are well understood in classical linear algebra, the implications of these properties in non-linear, stochastic optimization landscapes remain largely unexplored. In particular, questions remain about the representational capacity of networks that rely heavily on element-wise interactions, their convergence properties, and their generalization behavior [97]. Can networks with multiplicative interactions via Hadamard products approximate arbitrary functions under certain constraints [98]? What regularization effects do they implicitly impose? How do they compare, theoretically, to attention mechanisms or low-rank bilinear layers? These foundational questions represent a promising avenue for future research.

5.7 Summary

While the Hadamard product offers an elegant and efficient mechanism for local interactions in neural networks, its limitations—ranging from expressiveness and alignment constraints to optimization and interpretability challenges—highlight the need for deeper investigation. Addressing these issues will require a combination of theoretical analysis, empirical benchmarking, and low-level systems optimization [99]. As deep learning continues to evolve, refining our understanding of the Hadamard product and its variants will be essential for developing more robust, efficient, and interpretable models [100]. In the concluding section, we synthesize the insights gained and outline future directions for research on the Hadamard product in deep learning [101].

6 Conclusion and Future Directions

The Hadamard product, once considered a simple and peripheral mathematical operation, has emerged as a versatile and impactful mechanism within the modern deep learning landscape. From gating mechanisms in recurrent neural networks to attention modulation, feature fusion, and efficient conditioning of representations, the element-wise multiplication defined by $x \odot y$ has found wide-ranging utility across diverse neural architectures and tasks [102]. Its appeal lies in its computational efficiency, compatibility with gradient-based optimization, and natural alignment with the structure of high-dimensional data representations. In this survey, we began by examining the mathematical foundations and classical properties of the Hadamard product, situating it within the broader family of tensor operations. We then explored its applications in various deep learning architectures, including recurrent models, attention mechanisms, bilinear pooling, and meta-learning [103]. Our exploration of recent advances highlighted the creativity with which the research community has extended this operation—introducing learnable gates, probabilistic variants, log-space interactions, and generalized forms like trilinear or p-Hadamard products. These developments underscore the Hadamard product's continued relevance in enabling flexible, modular, and expressive neural computation. At the same time, we have drawn attention to the limitations and open challenges associated with the use of the Hadamard product [104]. Among these are issues related to its limited expressive power due to dimension-wise isolation, sensitivity to input scaling and gradient dynamics, and strict requirements for dimensionality matching [105]. Furthermore, although the operation facilitates intuitive feature selection and masking, its lack of theoretical underpinnings in learning dynamics, generalization, and representation expressivity remains a significant gap in the literature [106]. Looking ahead, several research directions appear promising:

- **Theoretical Characterization**: Developing formal frameworks to analyze the representational power and learning behavior of networks using the Hadamard product. This includes deriving approximation bounds, studying the interaction of multiplicative operations with non-linearities, and understanding their role in generalization and overfitting.
- **Efficient Architectures**: Designing neural models that combine the Hadamard product with structured projections, sparsity-inducing priors, or attention mechanisms to overcome its expressive limitations while preserving computational advantages [107].
- **Interpretability and Sparsity**: Creating more interpretable gating and masking strategies based on element-wise multiplication, potentially with structured regularization or explainability constraints, to support applications in safety-critical domains.
- **Cross-domain and Multimodal Learning**: Investigating how the Hadamard product can be used to align, fuse, or relate heterogeneous data types—such as text, images, graphs, or tabular data—through learned compatibility functions or modality-specific transformations [108].

- **Hardware-aware Optimization**: Exploring how to optimize the use of Hadamard products in hardware-accelerated environments, including GPU, TPU, or neuromorphic systems, through better memory layouts, kernel fusion, or compiler-level optimizations [109].
- **Extensions to Complex and Structured Domains**: Generalizing the Hadamard product to complex-valued tensors, non-Euclidean geometries, or structured domains such as manifolds, graphs, or quantum circuits, where multiplicative interactions may play a role in modeling entanglement or curvature [110].

In conclusion, the Hadamard product is no longer a mere computational footnote in the implementation of neural networks. Instead, it is a foundational primitive that invites further mathematical investigation, architectural creativity, and interdisciplinary application. As deep learning systems continue to grow in complexity and capability, understanding and leveraging such simple yet powerful operations will be key to building the next generation of intelligent models.

References

1. Zhen-Liang Ni, Gui-Bin Bian, Zhen Li, Xiao-Hu Zhou, Rui-Qi Li, and Zeng-Guang Hou. Space squeeze reasoning and low-rank bilinear feature fusion for surgical image segmentation. *IEEE Journal of Biomedical and Health Informatics*, 2022.
2. Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 207–212, 2016.
3. William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. pages 4960–4964, 2016.
4. Sung-Kwun Oh, Witold Pedrycz, and Byoung-Jun Park. Polynomial neural networks architecture: analysis and design. *Computers & Electrical Engineering*, 29(6):703–725, 2003.
5. Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. 2019.
6. Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2828–2837, 2019.
7. Xing Xu, Jialiang Sun, Zuo Cao, Yin Zhang, Xiaofeng Zhu, and Heng Tao Shen. Tfun: Trilinear fusion network for ternary image-text retrieval. *Information Fusion*, 91:327–337, 2023.
8. Hou-Biao Li, Ting-Zhu Huang, Shu-Qian Shen, and Hong Li. Lower bounds for the minimum eigenvalue of hadamard product of an m-matrix and its inverse. *Linear Algebra and its applications*, 420(1):235–247, 2007.
9. Pengfei Zhang, Jianru Xue, Cuiling Lan, Wenjun Zeng, Zhanning Gao, and Nanning Zheng. Eleatt-rnn: Adding attentiveness to neurons in recurrent neural networks. 29:1061–1073, 2019.

10. Xuezhe Ma, Chunting Zhou, Xiang Kong, Junxian He, Liangke Gui, Graham Neubig, Jonathan May, and Luke Zettlemoyer. Mega: moving average equipped gated attention. *arXiv preprint arXiv:2209.10655*, 2022.
11. Shuai Li, Wanqing Li, Chris Cook, Ce Zhu, and Yanbo Gao. Independently recurrent neural network (indrnn): Building a longer and deeper rnn. pages 5457–5466, 2018.
12. Weicong Kong, Zhao Yang Dong, Youwei Jia, David J Hill, Yan Xu, and Yuan Zhang. Short-term residential load forecasting based on lstm recurrent neural network. *IEEE Transactions on Smart Grid*, 10(1):841–851, 2017.
13. Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. volume 33, pages 922–929, 2019.
14. Cristian Rodriguez-Opazo, Edison Marrese-Taylor, Basura Fernando, Hongdong Li, and Stephen Gould. Dori: discovering object relationships for moment localization of a natural language query in a video. pages 1079–1088, 2021.
15. Fangneng Zhan, Yingchen Yu, Rongliang Wu, Jiahui Zhang, Kaiwen Cui, Aoran Xiao, Shijian Lu, and Chunyan Miao. Bi-level feature alignment for versatile image translation and manipulation. pages 224–241, 2022.
16. Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. 31(5):855–868, 2008.
17. Ming Liu, Yukang Ding, Min Xia, Xiao Liu, Errui Ding, Wangmeng Zuo, and Shilei Wen. Stgan: A unified selective transfer network for arbitrary image attribute editing. pages 3673–3682, 2019.
18. Thanh Le, Nam Le, and Bac Le. Knowledge graph embedding by relational rotation and complex convolution for link prediction. *Expert Systems with Applications*, 214:119122, 2023.
19. Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.
20. Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.
21. Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic memory networks for visual and textual question answering. pages 2397–2406, 2016.
22. David B Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. pages 16252–16262, 2022.
23. Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. pages 5505–5514, 2018.
24. Tan Nguyen, Richard Baraniuk, Andrea Bertozzi, Stanley Osher, and Bao Wang. Momentumrnn: Integrating momentum into recurrent neural networks. 33:1924–1936, 2020.
25. Yassine Zniyed, Thanh Phuong Nguyen, et al. Efficient tensor decomposition-based filter pruning. *Neural Networks*, 178:106393, 2024.
26. Junwen Wang, Shengwei Tian, Long Yu, Yongtao Wang, Fan Wang, and Zhicheng Zhou. Sbdf-net: A versatile dual-branch fusion network for medical image segmentation. *Biomedical Signal Processing and Control*, 78:103928, 2022.
27. Marshall H Stone. The generalized Weierstrass approximation theorem. *Math. Mag.*, 21(5):237–254, 1948.

28. Ricardo Augusto Borsoi, Tales Imbiriba, and José Carlos Moreira Bermudez. Super-resolution for hyperspectral and multispectral image fusion accounting for seasonal spectral variability. 29:116–127, 2019.
29. Yongtao Wu, Zhenyu Zhu, Fanghui Liu, Grigorios G Chrysos, and Volkan Cevher. Extrapolation and spectral bias of neural nets with hadamard product: a polynomial net study. 2022.
30. Kaiwen Tan, Weixian Huang, Xiaofeng Liu, Jinlong Hu, and Shoubin Dong. A multi-modal fusion framework based on multi-task correlation learning for cancer prognosis prediction. *Artificial Intelligence in Medicine*, 126:102260, 2022.
31. Weitao Jiang and Haifeng Hu. Hadamard product perceptron attention for image captioning. *Neural Processing Letters*, pages 1–18, 2022.
32. Jimmy Ba and Brendan Frey. Adaptive dropout for training deep neural networks. 26, 2013.
33. Yan Xiong, Wei Wu, Xidai Kang, and Chao Zhang. Training pi-sigma network by online gradient algorithm with penalty for small weight update. *Neural computation*, 19(12):3356–3368, 2007.
34. Hojoon Lee, Dongyoon Hwang, Sunghwan Hong, Changyeon Kim, Seungryong Kim, and Jaegul Choo. Moi-mixer: Improving mlp-mixer with multi order interactions in sequential recommendation. *arXiv preprint arXiv:2108.07505*, 2021.
35. Francesca Babiloni, Ioannis Marras, Filippos Kokkinos, Jiankang Deng, Grigorios Chrysos, and Stefanos Zafeiriou. Poly-nl: Linear complexity non-local layers with 3rd order polynomials. pages 10518–10528, 2021.
36. Aston Zhang, Yi Tay, Yikang Shen, Alvin Chan, and Shuai Zhang. Self-instantiated recurrent units with dynamic soft recursion. volume 34, pages 6503–6514, 2021.
37. Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. Light gated recurrent units for speech recognition. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(2):92–102, 2018.
38. Rong Dai, Li Shen, Fengxiang He, Xinmei Tian, and Dacheng Tao. Dispfl: Towards communication-efficient personalized federated learning via decentralized sparse training. 2022.
39. Shyamal Buch, Victor Escorcia, Chuanqi Shen, Bernard Ghanem, and Juan Carlos Niebles. Sst: Single-stream temporal action proposals. pages 2911–2920, 2017.
40. Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. Attention-based lstm for aspect-level sentiment classification. pages 606–615, 2016.
41. Jon R Kettenring. Canonical analysis of several sets of variables. *Biometrika*, 58(3):433–451, 1971.
42. Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. 2014.
43. Vishnu Suresh Lokhande, Songwong Tasneeyapant, Abhay Venkatesh, Sathya N Ravi, and Vikas Singh. Generating accurate pseudo-labels in semi-supervised learning and avoiding overconfident predictions via hermite polynomial activations. pages 11435–11443, 2020.
44. Diganta Misra. Mish: A self regularized non-monotonic neural activation function. *arXiv preprint arXiv:1908.08681*, 4(2):10–48550, 2019.
45. Yuan Cao, Zhiying Fang, Yue Wu, Ding-Xuan Zhou, and Quanquan Gu. Towards understanding the spectral bias of deep learning. 2021.
46. Rui Zhao, Dongzhe Wang, Ruqiang Yan, Kezhi Mao, Fei Shen, and Jinjiang Wang. Machine health monitoring using local feature-based gated recurrent unit networks. *IEEE Transactions on Industrial Electronics*, 65(2):1539–1548, 2017.

47. Tianshi Cao, Karsten Kreis, Sanja Fidler, Nicholas Sharp, and Kangxue Yin. Texfusion: Synthesizing 3d textures with text-guided image diffusion models. pages 4169–4181, 2023.
48. Yan Wang, Lingxi Xie, Chenxi Liu, Siyuan Qiao, Ya Zhang, Wenjun Zhang, Qi Tian, and Alan Yuille. Sort: Second-order response transform for visual recognition. pages 1359–1368, 2017.
49. Gourav Wadhwa, Abhinav Dhall, Subrahmanyam Murala, and Usman Tariq. Hyperrealistic image inpainting with hypergraphs. pages 3912–3921, 2021.
50. Syed Talal Wasim, Muhammad Uzair Khattak, Muzammal Naseer, Salman Khan, Mubarak Shah, and Fahad Shahbaz Khan. Video-focalnets: Spatio-temporal focal modulation for video action recognition. pages 13778–13789, 2023.
51. Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.
52. David So, Wojciech Mańke, Hanxiao Liu, Zihang Dai, Noam Shazeer, and Quoc V Le. Searching for efficient transformers for language modeling. volume 34, pages 6010–6022, 2021.
53. T Konstantin Rusch and Siddhartha Mishra. Unicornn: A recurrent model for learning very long time dependencies. pages 9168–9178, 2021.
54. Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. 2018.
55. Li-Ming Zhan, Bo Liu, Lu Fan, Jiaxin Chen, and Xiao-Ming Wu. Medical visual question answering via conditional reasoning. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2345–2354, 2020.
56. Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. volume 32, 2019.
57. Haiwei Pan, Shuning He, Kejia Zhang, Bo Qu, Chunling Chen, and Kun Shi. Amam: An attention-based multimodal alignment model for medical visual question answering. *Knowledge-Based Systems*, 255:109763, 2022.
58. Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. Graphrnn: Generating realistic graphs with deep auto-regressive models. pages 5708–5717, 2018.
59. Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of detr with spatially modulated co-attention. pages 3621–3630, 2021.
60. Mauro Dalla Mura, Saurabh Prasad, Fabio Pacifici, Paulo Gamba, Jocelyn Chanussot, and Jón Atli Benediktsson. Challenges and opportunities of multimodality and data fusion in remote sensing. *Proceedings of the IEEE*, 103(9):1585–1601, 2015.
61. Aming Wu, Linchao Zhu, Yahong Han, and Yi Yang. Connective cognition network for directional visual commonsense reasoning. 32, 2019.
62. Hyeonseob Nam, Jung-Woo Ha, and Jeonghee Kim. Dual attention networks for multimodal reasoning and matching. pages 299–307, 2017.
63. Wenting Chen, Yifan Liu, Jiancong Hu, and Yixuan Yuan. Dynamic depth-aware network for endoscopy super-resolution. *IEEE Journal of Biomedical and Health Informatics*, 26(10):5189–5200, 2022.
64. Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014.

65. Fangfang Yang, Weihua Li, Chuan Li, and Qiang Miao. State-of-charge estimation of lithium-ion batteries based on gated recurrent neural network. *Energy*, 175:66–75, 2019.
66. Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. pages 315–323, 2011.
67. Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. 2017.
68. Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. pages 7008–7024, 2017.
69. C Lee Giles and Tom Maxwell. Learning, invariance, and generalization in high-order neural networks. *Applied optics*, 26(23):4972–4978, 1987.
70. Peng Xu, Xiatian Zhu, and David A Clifton. Multimodal learning with transformers: A survey. 45(10):12113–12132, 2023.
71. Zequn Qin, Pengyi Zhang, Fei Wu, and Xi Li. Fcanet: Frequency channel attention networks. pages 783–792, 2021.
72. Junlong Cheng, Shengwei Tian, Long Yu, Hongchun Lu, and Xiaoyi Lv. Fully convolutional attention network for biomedical image segmentation. *Artificial Intelligence in Medicine*, 107:101899, 2020.
73. Xu Yang, Chongyang Gao, Hanwang Zhang, and Jianfei Cai. Auto-parsing network for image captioning and visual question answering. pages 2197–2207, 2021.
74. Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? pages 8296–8305, 2020.
75. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. pages 5998–6008, 2017.
76. Kaisei Fukaya, Damon Daylamani-Zad, and Harry Agius. Evaluation metrics for intelligent generation of graphical game assets: a systematic survey-based framework. 2024.
77. Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. 2017.
78. Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
79. Chi Zhang, Guosheng Lin, Lvlong Lai, Henghui Ding, and Qingyao Wu. Calibrating class activation maps for long-tailed visual recognition. *arXiv preprint arXiv:2108.12757*, 2021.
80. Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. 2019.
81. Chaohao Xie, Shaohui Liu, Chao Li, Ming-Ming Cheng, Wangmeng Zuo, Xiao Liu, Shilei Wen, and Errui Ding. Image inpainting with learnable bidirectional attention maps. pages 8858–8867, 2019.
82. Peng Gao, Zhengkai Jiang, Haoxuan You, Pan Lu, Steven CH Hoi, Xiaogang Wang, and Hongsheng Li. Dynamic fusion with intra-and inter-modality attention flow for visual question answering. pages 6639–6648, 2019.
83. Eric David Petajan. *Automatic lipreading to enhance speech recognition (speech reading)*. PhD thesis, 1984.
84. Xuan Chen, Jun Hao Liew, Wei Xiong, Chee-Kong Chui, and Sim-Heng Ong. Focus, segment and erase: an efficient network for multi-label brain tumor segmentation. pages 654–669, 2018.

85. Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xlstm: Extended long short-term memory. 2024.
86. Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610, 2005.
87. Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, and et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
88. Jireh Jam, Connah Kendrick, Vincent Drouard, Kevin Walker, Gee-Sern Hsu, and Moi Hoon Yap. R-mnet: A perceptual adversarial network for image inpainting. pages 2714–2723, 2021.
89. Difei Gao, Ke Li, Ruiping Wang, Shiguang Shan, and Xilin Chen. Multi-modal graph neural network for joint reasoning on vision and scene text. pages 12746–12756, 2020.
90. Jssai Schur. Bemerkungen zur theorie der beschränkten bilinearformen mit unendlich vielen veränderlichen. 1911.
91. Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. pages 173–182, 2016.
92. Chen Ma, Peng Kang, and Xue Liu. Hierarchical gating networks for sequential recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 825–833, 2019.
93. Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.
94. Ben P Yuhas, Moise H Goldstein, and Terrence J Sejnowski. Integration of acoustic and visual speech signals using neural networks. *Communications Magazine*, 27(11):65–71, 1989.
95. Yassine Zniyed, Thanh Phuong Nguyen, et al. Enhanced network compression through tensor decompositions and pruning. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.
96. Zhen-Liang Ni, Gui-Bin Bian, Guan-An Wang, Xiao-Hu Zhou, Zeng-Guang Hou, Hua-Bin Chen, and Xiao-Liang Xie. Pyramid attention aggregation network for semantic segmentation of surgical instruments. volume 34, pages 11782–11790, 2020.
97. Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.
98. Teng-Yu Ji, Delin Chu, Xi-Le Zhao, and Danfeng Hong. A unified framework of cloud detection and removal based on low-rank and group sparse regularizations for multitemporal multispectral images. 60:1–15, 2022.
99. Grigorios G Chrysos, Stylianos Moschoglou, Giorgos Bouritsas, Yannis Panagakis, Jiankang Deng, and Stefanos Zafeiriou. $\pi-$nets: Deep polynomial neural networks. pages 7325–7335, 2020.
100. Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. pages 807–814, 2010.
101. Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. 2024.
102. Jianwei Yang, Anitha Kannan, Dhruv Batra, and Devi Parikh. Lr-gan: Layered recursive generative adversarial networks for image generation. 2017.

103. Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. Quaternion knowledge graph embeddings. 32, 2019.
104. Renlong Hang, Qingshan Liu, Danfeng Hong, and Pedram Ghamisi. Cascaded recurrent neural networks for hyperspectral image classification. 57(8):5384–5394, 2019.
105. Kazihise Ntikurako Guy-Fernand, JuanJuan Zhao, Fredy Malack Sabuni, and Jiawen Wang. Classification of brain tumor leveraging goal-driven visual attention with the support of transfer learning. In *2020 Information Communication Technologies Conference (ICTC)*, pages 328–332. IEEE, 2020.
106. Julian Jorge Andrade Guerreiro, Naoto Inoue, Kento Masui, Mayu Otani, and Hideki Nakayama. Layoutflow: flow matching for layout generation. pages 56–72. Springer, 2024.
107. Hanxiao Liu, Zihang Dai, David So, and Quoc V Le. Pay attention to mlps. 34:9204–9215, 2021.
108. Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. pages 2337–2346, 2019.
109. George PH Styan. Hadamard products and multivariate statistical analysis. *Linear algebra and its applications*, 6:217–240, 1973.
110. Chao Ma, Chunhua Shen, Anthony Dick, Qi Wu, Peng Wang, Anton van den Hengel, and Ian Reid. Visual question answering with memory-augmented networks. pages 6975–6984, 2018.