DISSERTATION

Domain Obedient Deep Learning

Soumadeep Saha

A dissertation submitted in partial fulfillment of the requirements for the degree of *Doctor of Philosophy* in Computer Science.

Supervisor: Prof. Utpal Garain

Computer Vision and Pattern Recognition Unit
Indian Statistical Institute, Kolkata

July, 2025

to my late grandfather
Ashutosh Sadhu

Declaration of Authorship

I, *Soumadeep Saha*, declare that this dissertation, titled "Domain Obedient Deep Learning", and the work presented in it, are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this university.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this university or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Soumadeep Saha
15/07/2025

Abstract

Deep learning, a family of data-driven artificial intelligence techniques, has shown immense promise in a plethora of applications, and it has even outpaced experts in several domains. However, unlike symbolic approaches to learning, these methods fall short when it comes to abiding by and learning from pre-existing established principles. This is a significant deficit for deployment in critical applications such as robotics, medicine, industrial automation, etc. For a decision system to be considered for adoption in such fields, it must demonstrate the ability to adhere to specified constraints, an ability missing in deep learning-based approaches. Exploring this problem serves as the core tenet of this dissertation.

This dissertation starts with an exploration of the abilities of conventional deep learning-based systems *vis-à-vis* domain coherence. A large-scale rule-annotated dataset is introduced to mitigate the pronounced lack of suitable constraint adherence evaluation benchmarks, and with its aid, the rule adherence abilities of vision systems are analyzed. Additionally, this study probes language models to elicit their performance characteristics with regard to domain consistency. Examination of these language models with interventions illustrates their ineptitude at obeying domain principles, and a mitigation strategy is proposed. This is followed by an exploration of techniques for imbuing deep learning systems with domain constraint information. Also, a comprehensive study of standard evaluation metrics and their blind spots pertaining to domain-aware performance estimation is undertaken. Finally, a novel technique to enforce constraint compliance in models without training is introduced, which pairs a search strategy with large language models to achieve cutting-edge performance.

A key highlight of this dissertation is the emphasis on addressing pertinent real-world problems with scalable and practicable solutions. We hope the results presented here pave the way for wider adoption of deep learning-based systems in pivotal situations with enhanced confidence in their trustworthiness.

Acknowledgements

Embarking on this doctoral journey, I truly felt like a fish out of water, but over time I discovered a fascinating world full of remarkable possibilities that captured my imagination and fueled my passion for inquiry. Completing this journey has been profoundly rewarding, a milestone made possible by the unwavering support and encouragement of many remarkable individuals.

First and foremost, I would like to extend my heartfelt gratitude to my advisor, Prof. Utpal Garain. Your invaluable guidance, expertise, and patience have been instrumental in every facet of my research. Your belief in my capabilities has been a profound source of motivation. I am incredibly fortunate to have had a mentor of your caliber.

I am forever indebted to my parents, whose unconditional love and steadfast encouragement have been a bedrock of support throughout my academic endeavors. Your sacrifices and faith in my potential have been the driving force behind my pursuit of this academic milestone. Thank you for always believing in me.

I extend a special thanks to Akshay Chaturvedi for showing me the ropes. I am also grateful to my colleagues and friends - Joy Mahapatra, Saptarshi Saha, and Aditya Shankar Pal who have offered their support and encouragement throughout this process. I have been immensely blessed with the opportunity to collaborate with exceptional individuals, and I extend my sincere thanks to Nicholas Asher, Supratik Pal, Rahul Shah, Purba Mukherjee, Arijit Ukil, Arpan Pal, Claudia Stern, Scott Ritter, and Aude Benk-Fortin for being a part of this journey and for their valuable insights.

I am deeply grateful to my institute for providing this opportunity to pursue my doctoral studies, and for fostering a community of nurturing individuals who have played a crucial and supportive role in my academic journey. I cherish the memories and experiences gained within this esteemed institution.

Lastly to my partner, Sutanoya, your understanding, affection, and unwavering support have been my anchor during the highs and lows of this journey. Having an in-house world-class mathematician and newspaper-worthy copy editor is an incredibly valuable resource, especially when it only comes at the cost of an equal share of my caloric intake. Jokes aside, your presence has been a source of strength and comfort, making this journey all the more meaningful.

■

CONTENTS

List of Figures

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Meaning
SoTA	State-of-the-art
AI	Artificial intelligence
ML	Machine learning
DL	Deep learning
CV	Computer vision
NLP	Natural language processing
LM	Language model
LLM	Large language model
NN	Neural network
FC	Fully connected
MLP	Multi-layer perceptron
RNN	Recurrent neural network
LSTM	Long short term memory network
CNN	Convolutional neural network
MLC	Multi-label classification
MCC	Multi-class classification
QA	Question answering
MSE	Mean-squared error
CE	Cross-entropy
BCE	Binary cross-entropy

“Turing believes machines think
Turing lies with men
Therefore machines do not think”

– Alan Turing

1 INTRODUCTION

Historically, artificial intelligence (AI) research has followed two disparate development paths. One path, termed *symbolic* AI, represents knowledge in human-readable symbols and performs reasoning by manipulating the symbols and applying rules of logical inference. The other path is *data-centric* AI, the inarguably more successful sibling, which has been getting the biggest share of attention lately. Although there are a plethora of techniques that fall under the umbrella of *data-centric* AI or machine learning (ML) like support vector machines [23], nearest neighbours [47] or the humble linear regression, one of the most successful techniques in ML has been deep learning (DL).

DL techniques have been widely applied to a multitude of domains and have dethroned expert-driven specialized systems to become ubiquitous in many areas of computer vision (CV), natural language processing (NLP), biotechnology, etc. The core ideas promulgating DL today are not new, and prototype techniques have been proposed since the 1960s [140, 145]. The current boom in DL applications can be largely attributed to the availability of massive amounts of data and computational power. IBM introduced the 1301 disk storage unit [78] not long after Rosenblatt [140] conceptualized the first prototype neural network (NN), and today large language models (LLMs), a product of frontier DL research, are trained on the equivalent of two million such drives worth of data while using the same amount of power as the Republic of Vanuatu does in a month.

Despite the undeniable triumph of DL techniques, *symbolic* AI techniques have some appealing features, the most notable of which is their ability to leverage and adhere to established domain knowledge and rules. With the widespread use of DL, it has become apparent that in some situations, this ability is indispensable, and conventional DL techniques demonstrate a dire shortcoming in this regard. This lack of domain knowledge adherence marring an incredibly promising family of techniques is the focus of this dissertation. A hybrid approach learning from data alongside domain principles and providing decisions in line with problem-specific constraints would be remarkably opportune.

In this chapter we first introduce the necessary background *vis à vis* deep learning and associated techniques (Sections 1.1 and 1.2), before briefly discussing some trends in attempts to incorporate domain knowledge adherence in DL systems (Section 1.3). Section 1.4 lays out the organization of the rest of the dissertation.

1.1 Deep Learning

Deep learning (DL) refers to a powerful and diverse family of techniques that leverage data to model complex real world phenomena. What sets apart DL from the majority of other techniques in this regime is its unparalleled versatility. DL has been successfully applied to, and attained state-of-the-art (SoTA) results in, domains like CV, NLP, reinforcement learning, biotechnology [86], medicine [63], and hundreds more.

Although DL features multifarious techniques, architectures, training paradigms, and methodologies, there are a few unifying factors. First and most important of which is *data*, and lots of it. This data can be un-annotated *(unsupervised)* or annotated by crowdsourcing or appointing experts *(supervised)*. Also, available data is often cleverly repurposed by making assumptions or employing additional techniques *(weakly supervised)*. Secondly, DL-based techniques feature at least one parametrized non-linear almost-everywhere differentiable function (*model*), which is meant to imitate the underlying relationship we wish to explore. Finally, they feature an objective or *loss* function, quantifying desired outcomes for the model when applied to the dataset in question. With just these three ingredients, a large array of problems in varied areas can be tackled.

In this section, we go over some of the background necessary for the rest of the

presentation. We do not provide a thorough overview of DL, and we recommend the work by Goodfellow et al. [60] for an expansive treatment.

1.1.1. BASIC FORMALISM

NOTATION

A dataset ($\mathcal{D}$) is typically a set of ordered k-tuples, with $k \in \{1, 2\}$. The $k = 1$ case refers to the unsupervised setting, and we can simply write $\mathcal{D} = \{x_i \,|\forall i \in \{1, \ldots, n\}\}$, where n is the size of the dataset ($|\mathcal{D}| = n$), and the x_is are sampled from the underlying distribution we wish to study. Although not strictly necessary, typically $x_i \in \mathbb{R}^d$ for some $d \in \mathbb{N}$, and is called the *input*. The $k = 2$ case refers to the supervised setting, and we have $\mathcal{D} = \{(x_i, y_i) \mid \forall i \in \{1, \ldots, n\}\}$. The y_is are called the *ground truths*, *labels*, or *annotations* and can refer to several objects like finite sets, *one-hot* vectors, real numbers, real-valued vectors, etc., and serve as the desired association corresponding to the input x_i.

The model is represented with $f, g, h \ldots$ and typically appears with a subscript - f_θ. f_θ is a function, and we have:

$$f_\theta : \Lambda \to \mathbb{R}^D \text{ for some } D \in \mathbb{N} \tag{1.1}$$

where Λ is the space from which the input samples are drawn ($x_i \in \Lambda$). The vector $\hat{y}_i = f_\theta(x_i)$ is termed the *output* of the model corresponding to input x_i. θ is the set of all parameters required for the computation of the function, and typically:

$$\begin{aligned} &\theta = \{W_1, W_2, \ldots, b_1, b_2, \ldots, \text{etc.}\} \\ &W_i \in \mathbb{R}^{M \times N} \text{ are matrices. } b_i \in \mathbb{R}^K \end{aligned} \tag{1.2}$$

If θ is subject to evolution, we denote the set of parameters at time t with θ_t, and if the "optimal" value is attained by all parameters in θ, we refer to the set as θ^*. The *model* f_θ is continuous and at least piecewise differentiable with respect to the parameters in θ. The objective or *loss function* is denoted with $\mathcal{L}$ or ℓ.

We often need to talk about a particular index of a vector or matrix, and we refer to the i^{th} index of a vector v and the i, j^{th} index of matrix M as $[v]_i$ and $[M]_{ij}$, respectively. For a real-valued function $g : \mathbb{R} \to \mathbb{R}$, we often use the shorthand notation $g(v),\ v \in \mathbb{R}^\alpha$, to represent the vector whose i^{th} component is $g([v]_i)$.

FOUNDATIONAL TECHNIQUES

Given a dataset $\mathcal{D}$, a model f_θ, and a loss function $\mathcal{L}$, such that

$$\begin{aligned} \mathcal{L} :&\mathbb{R}^D \times \ldots \rightarrow \mathbb{R}^+ \cup \{0\} \\ &\mathcal{L}(f_\theta(x_i), \ldots) \mapsto s_i \end{aligned} \tag{1.3}$$

where x_i is an instance in $\mathcal{D}$, and s_i can be interpreted as the *"undesirability"* of x_i taking on the value $\hat{y}_i = f_\theta(x_i)$; DL lays out a framework for finding values of the parameters in θ:

$$\theta^* = \arg\min_\theta \mathbb{E}_{x\sim\mathcal{D}}\Big[\mathcal{L}\big(f_\theta(x), \ldots\big)\Big] \tag{1.4}$$

such that they are optimal (*local minima*). [1] This hinges on two key ideas–*gradient descent* and *backpropagation* [103].

Gradient descent exploits the following fact:

$$\begin{aligned} &\nabla_W \mathcal{L} \cdot h = \mathcal{L}(W + h) - \mathcal{L}(W) + \mathcal{O}(h^2) && \text{by definition.} \\ &\text{Setting } h = -\eta\nabla_W\mathcal{L},\ \nabla_W\mathcal{L} \cdot h = -\eta||\nabla_W\mathcal{L}||^2 \leq 0 \\ &\Rightarrow \mathcal{L}(W - \eta\nabla_W\mathcal{L}) - \mathcal{L}(W) + \mathcal{O}(h^2) \leq 0 \\ &\Rightarrow \mathcal{L}(W - \eta\nabla_W\mathcal{L}) \leq \mathcal{L}(W) && \text{for some appropriately small } \eta > 0. \end{aligned}$$

i.e., around the value of a parameter $W \in \theta$, there exists a direction $(-\nabla_W\mathcal{L})$ along which we can perturb W by a small amount (η), such that the loss (undesirability) reduces. The step size η is called the *learning rate* and is a *hyperparameter* that must be externally set. This fact can be used to devise an iterative optimization algorithm with the update rule:

$$W_t^i = W_{t-1}^i - \eta\frac{1}{B}\sum_{i=1}^{B}\nabla_{W_{t-1}^i}\mathcal{L}(f_{\theta_{t-1}}(x_i), \ldots) \qquad \forall\, W^i \in \theta \tag{1.5}$$

This algorithm is commonly referred to as *mini-batch gradient descent* or *stochastic gradient descent* (SGD) [142]. The value B, called the *batch size* is another hyperparameter, and B samples from the dataset are randomly drawn to estimate $\mathbb{E}[\mathcal{L}]$. The update stops at stationary points, i.e., where $\nabla_W\mathcal{L} = 0$, and we use the final set

[1] The exact form and inputs of the function $\mathcal{L}$ are dependent on the task, and we outline a few examples in the upcoming section.

of parameters found as our predictive model f_{θ^*}. *Adam* [91], a refinement of SGD, employing adaptive learning rate scaling and momentum, is more versatile and is in widespread use today.

Backpropagation [103, 145] is a technique to efficiently calculate the required gradients $\nabla_W \mathcal{L}(f_\theta(x), \ldots)$. The gradients are calculated by evaluating the closed-form partial derivatives of the functions using their explicit formulas and the chain rule. The model f_θ is expressed as a directed (acyclic) graph where each vertex is an input, parameter, or a function, and there is an edge from node m to node n if m is an input to the function represented by node n. Along a path $a_1 \to a_2 \to \ldots \to a_n$ in the graph, to calculate $\frac{\partial a_n}{\partial a_k}$ we can compute $\frac{\partial a_n}{\partial a_{k+1}}$ and $\frac{\partial a_{k+1}}{\partial a_k}$, and combine using the chain rule. The backpropagation algorithm avoids repeated gradient computations by performing them in reverse topologically sorted order.

Thus, to summarize, given f_θ, $\mathcal{D}$, and $\mathcal{L}$, we sample a batch (B) of inputs from $\mathcal{D}$, compute $L = \sum_{i=1}^{B} \mathcal{L}(f_\theta(x_i), \ldots)$ followed by $\nabla_W L$ for every $W \in \theta$ using the backpropagation algorithm. We then update the parameters in θ with the SGD (or Adam, etc.) update rule and repeat this process until some convergence criteria is met. This is a common approach to finding a suitable model across a wide range of applications in DL.

EXAMPLE USES

DL is used across a variety of settings with different datasets, models, and objective functions, ranging from language modeling, semantic segmentation, variational auto-encoders, and denoising-diffusion to contrastive learning, reinforcement learning, meta-learning, and more. We provide a few illustrative example tasks and their associated objective functions here.

In non-linear *regression* tasks, $\mathcal{D} = \{(x_i, y_i) | \forall i \in \{1, \ldots, n\}\}$, where $x_i \in \mathbb{R}^M$ and $\mathbb{R}^N$. The loss function is typically L_2 norm aka mean-squared error (MSE) of the output $\hat{y}_i$ and ground truth y_i, although other disparate losses like smoothL_1 or *intersection-over-union* (bounding box regression) are often used.

In *binary classification*, every sample $(x_i, y_i) \in \mathcal{D}$ belongs to the '$+$' or '$-$' class and has an associated $y_i = P(+|x_i)$. The model's output $\hat{y}_i = f_\theta(x_i) \in (0, 1)$ is interpreted as the probability $Q(+|x_i) := \hat{y}_i$. The typical loss function in this setting

is the binary cross-entropy (BCE) loss defined as:

$$\begin{aligned}\mathcal{L}_{\text{BCE}}(\hat{y}_i, y_i) &= H(P,Q) = -\mathbb{E}_P[log(Q)] \\ &= -P(+|x_i)\log\big(Q(+|x_i)\big) - \big(1-P(+|x_i)\big)\log\big(1-Q(+|x_i)\big) \\ &= -P(+|x_i)\log\big(Q(+|x_i)\big) - P(-|x_i)\log\big(Q(-|x_i)\big)\end{aligned} \tag{1.6}$$

If discrete decisions are needed, we fix a constant $\kappa \in (0,1)$, such that x_i is '+' if $\hat{y}_i \geq \kappa$.

Multi-class classification (MCC) generalizes this notion to C classes, and for every x_i we have $y_i \in (0,1)^C$ such that $[y_i]_\mu = P(\mu|x_i)$ and $\sum_{\nu=1}^{C}[y_i]_\nu = 1$. Here we typically use the general cross-entropy (CE) loss defined as:

$$\begin{aligned}\mathcal{L}_{\text{CE}}(\hat{y}_i, y_i) &= H(P,Q) = -\mathbb{E}_P[log(Q)] \\ &= -\sum_{\nu=1}^{C} P(\nu|x_i)\log\big([\hat{y}_i]_\nu\big)\end{aligned} \tag{1.7}$$

Auto-encoders employ a model f_θ to create a low-dimensional representation $z_i \in \mathbb{R}^d$ for $x_i \in \mathbb{R}^D$ ($d < D$) and another model g_ϕ to reconstruct x_i from z_i. The combined model $(f \circ g)_{\{\theta,\phi\}}$ uses a reconstruction loss like MSE between x_i and $(f \circ g)_{\{\theta,\phi\}}(x_i)$.

1.1.2. POPULAR MODELS

After the concise look at the methodology of DL, we turn to the models, which are in this context neural networks (NNs). This terminology harkens back to the origins of the study of these models, which were first proposed as a facsimile of the brain. These models are typically built up from a composition of simple *building blocks*, and we explore a few popular designs in this section.

MULTI-LAYER PERCEPTRON (MLP)

DEFINITION 1.1 (MLP [152]) *An MLP is defined by the equation:*

$$\begin{aligned}h_l &= \sigma(W_l h_{l-1} + b_l) \\ \hat{y}_i &= f_\theta(x_i) := h_L;\ h_0 = x_i\end{aligned}$$

where h_l is the latent output at layer *l, computed by multiplying the output of layer $l-1$ with a **weight** W_l and adding a **bias** b_l, before applying a non-linear **activation***

*or **transfer** function σ (W_l is a linear map and b_l is a vector).*

Without the non-linear activation function, the MLP model reduces to a linear model, and thus the former is a crucial component of building NNs. Although simple in construction, this model with an appropriately chosen activation function has powerful properties.

THEOREM 1.1 (Universal Function Approximation [25]) [arbitrary width case] *$\sigma \in \mathcal{C}(\mathbb{R}, \mathbb{R})$ is not a polynomial if and only if for $n, m \in \mathbb{N}$, a compact subset $K \subseteq \mathbb{R}^n$, $f \in \mathcal{C}(K, \mathbb{R}^m)$ and $\epsilon > 0$ $\exists A \in \mathbb{R}^{k \times n}, b \in \mathbb{R}^k, C \in \mathbb{R}^{m \times k}$ such that*

$$\sup_{x \in K} ||f(x) - C \cdot \sigma(Ax + b)|| < \epsilon$$

Theorem 1.1 basically states that an MLP with a single[2] hidden layer can approximate arbitrary functions if equipped with a non-polynomial activation function. Variants of this theorem with a bounded width and arbitrary depth also exist [109], and these results demonstrate the potent expressiveness of MLPs. Popular choices for activation functions include the *sigmoid* defined as $\sigma(x) = 1/(1 + \exp(-x))$, *ReLU* defined $\text{ReLU}(x) = x$ if $x \geq 0$ else 0, $\tanh$, and GeLU [66] defined as:

$$\text{GeLU}(x) = x \cdot \frac{1}{2}\Big[1 + \text{erf}(x/\sqrt{2})\Big] \tag{1.8}$$

Deep MLPs, also called fully connected (FC) networks, have been used across a wide range of applications, and they continue to play a pivotal role in SoTA architectures [183].

CONVOLUTIONAL NEURAL NETWORK (CNN)

CNNs, introduced by LeCun et al. [100], see extensive applications in image tasks [94] and are characterized by the convolution operation.

DEFINITION 1.2 (Convolution) *Given an image $A \in \mathbb{R}^{H \times W \times C}$ and a kernel $K \in \mathbb{R}^{(2m+1) \times (2n+1) \times C}$, $H, W, C, m, n \in \mathbb{N}$ we have:*

$$[Conv(A, K)]_{i,j} = \sum_{a=-m}^{m} \sum_{b=-n}^{n} \sum_{c=1}^{C} [A]_{i+a,j+b,c} \cdot [K]_{a+m+1,b+n+1,c}$$

[2] potentially infinitely wide.

for $m < i < H - m$ *and* $n < j < W - n$. [3]

A CNN uses several such groups of convolution operations with learnable parameters stacked on top of each other, with activation functions and other operations like pooling, etc. [94] in between. The final stages of CNNs are typically FC networks. Simonyan and Zisserman [167] introduced VGG-19, a "very" deep[4] version of CNNs to tackle the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [28].

Extremely deep networks face significant training difficulties owing to diminished gradients for initial layers, thus limiting the depth of early CNNs. However, the introduction of *residual* (or *skip* or *shortcut*) connections has allowed for the instantiation of much deeper CNN-based networks [65]. For a model $(f \circ g \circ h)(x)$, adding residual connections would transform it to the form $(f \circ g \circ h + f \circ h)(x)$ and allow gradient flow to h despite potential adverse effects of g. The ResNet architecture [65] increased depth by an order of magnitude and beat previous benchmarks on ILSVRC.

Another notable property of DL-based NNs is their *transfer learning* ability. It has been observed that large-scale training on low-quality supervised or unsupervised datasets *(pre-training)* followed by further training on smaller, higher-quality datasets *(fine-tuning)* yields performance benefits for the downstream task and is a widely used paradigm for a variety of tasks [222].

SEQUENCE MODELS

When the application calls for modeling sequences $\{x_i\}, x_i \in \mathbb{R}^N$ we often resort to the recurrent neural network (RNN) architecture.

DEFINITION 1.3 (RNN [144]) *Given* $h_0 \in \mathbb{R}^M$, $W \in \mathbb{R}^{M \times N}$, $V \in \mathbb{R}^{M \times M}$, $U \in \mathbb{R}^{D \times M}$ *and* $b_h, b_y \in \mathbb{R}^M, \mathbb{R}^D$ *respectively, the following relations:*

$$h_t = \sigma(Wx_t + Vh_{t-1} + b_h)$$
$$y_t = \sigma(Uh_t + b_y) \qquad \sigma(x) = 1/(1 + \exp(-x))$$

define an RNN.

This network can be used in different ways depending on the task at hand, like

[3] Borders can be set to zero, removed, etc.
[4] 19 layers of convolutions.

mapping a sequence to a vector, a vector to a sequence, or a sequence to another sequence. The weights W, V, U, etc., are shared between time steps, and this model is trained with the *backpropagation through time* (BPTT) [198] algorithm, which involves *"unrolling"* the network along a (finite) sequence, followed by performing backpropagation for each time step and adding their contributions. RNNs have significant limitations with regard to sequence length and face the *vanishing / exploding gradients problem*, i.e. the gradient for time-step t has a W^t term. To allay this, long short term memory networks (LSTMs) [72] were introduced.

DEFINITION 1.4 (LSTM [72]) *Given $h_0 \in \mathbb{R}^N$ and a sequence of input vectors $(x_1, x_2, \ldots)$*

$$\begin{aligned} f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\ \tilde{c}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \tag{1.9}$$

defines an LSTM network, where W_*, U_* are matrices, b_* are bias vectors, and $\odot$ represents the Hadamard product.

TRANSFORMERS

The *transformer* model, proposed by Vaswani et al. [183], has achieved SoTA performance across a variety of tasks in NLP, CV, etc., by enabling greater scalability owing to its parallelizable design. Although it was first proposed in the context of NLP tasks, it is now the model of choice for a plethora of other tasks [34, 86, 92]. There are subtle differences in the architecture in different use cases, e.g., creating patches of images for CV tasks [34], etc., and we only go over the primary NLP variant in this section. Figure 1.1 contains a pictorial representation of the transformer encoder/decoder block[5].

The dataset used to train transformers in the NLP setting is copious amounts of unstructured plain text, and in order to train one, the first step is training a *tokenizer* model. This model breaks up text into chunks called *tokens*, based on occurrence

[5]Conventionally an encoder-only or decoder-only transformer is used, and the only difference between them is in masking. The decoder block is different when used in an encoder-decoder configuration.

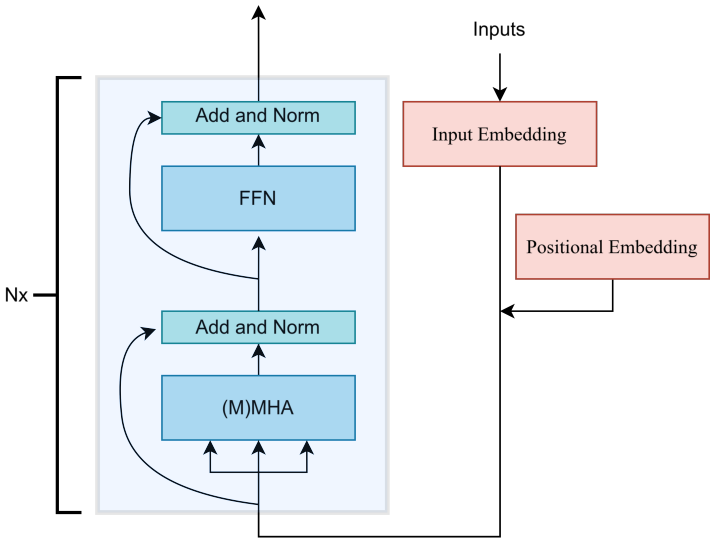


FIGURE 1.1: Diagram of transformer [183] model.

frequency, so that (a) any piece of text can be written as a combination of these tokens, (b) there are no extremely rare tokens, and (c) we have a fixed finite number (V) of tokens. Once a trained tokenizer is obtained (byte-pair encoding [159] for example), any piece of text can be converted to a sequence of tokens and fed to a transformer.

The input embedding step (see Figure 1.1) can be thought of as a lookup table, mapping each token t to a vector $v_t \in \mathbb{R}^d$, and the positional embedding provides a vectorial representation of the position of the token, which is added to the input embedding. There are several choices for the positional embedding scheme, like *sinusoidal* [183], *learned positional embedding* [34], or RoPE [172].

Following this step, we proceed to the first transformer encoder/decoder block, and typically there are several such identical blocks stacked together to form the model. In each block we have the *multi-head self-attention* (**MHA**) or the *masked multi-head self-attention* (**MMHA**) block for the encoder or decoder, respectively.

DEFINITION 1.5 (MHA) *Given $H \in \mathbb{N}$ (number of heads) and input sequence*

$(v_1, v_2, \dots v_T)$, $v_j \in \mathbb{R}^d \, \forall j, 1 \leq j \leq T$:

$$
\begin{aligned}
Q_j^i, K_j^i, V_j^i &:= W_{Q^i} \cdot x_j, W_{K^i} \cdot x_j, W_{V^i} \cdot x_j \\
&\qquad \forall i, j \textit{ such that } 1 \leq i \leq H, 1 \leq j \leq T \\
A_j^i &:= \sum_{l=1}^{T} \textit{softmax}\Big(\frac{Q_j^i \cdot K_l^i}{\sqrt{d}}\Big) \cdot V_l^i && \text{(self-attention)} \\
\hat{y}_j &:= W_O \cdot \textit{concat}(A_j^1, A_j^2, \dots A_j^H) && \textbf{(Output)}
\end{aligned}
$$

defines an MHA layer. $W_Q, W_K, W_V \in \mathbb{R}^{d/H \times d}$, $W_O \in \mathbb{R}^{d \times d}$ *are matrices, and softmax is defined as:*

$$
[\textit{softmax}(v)]_i := \frac{\exp([v]_i)}{\sum_{j=1}^{d} \exp([v]_j)}
$$

For **MMHA** the only difference is in the self-attention:

$$
A_j^i := \sum_{\substack{l=1 \\ l \leq j}}^{T} \text{softmax}\Big(\frac{Q_j^i \cdot K_l^i}{\sqrt{d}}\Big) \cdot V_l^i \qquad \text{(\textbf{MMHA} self-attention)}
$$

This is done to ensure that information from future tokens, which would not be available during inference, is not used during training (*causal masking*).

The **FFN** (see Figure 1.1) consists of two FC layers, mapping $v_i \in \mathbb{R}^d \to \mathbb{R}^{\alpha d} \to \mathbb{R}^d$ with GeLU activation in between. The **Add and Norm** consists of a *shortcut* connection and *LayerNorm* normalization [183].

1.2 LANGUAGE MODELS

Transformer-based language models (LMs) have revolutionized the field of NLP and have had an outsized impact on everyday life, with powerful LLMs like GPT-4 [126], Llama 3 [116], and Gemini [177] powering omnipresent services like conversational chat and internet search. In this section we look at some cursory details for LMs.

DEFINITION 1.6 (language model) *A function g is called a language model if*

given any string of tokens $\mathbf{x} = (x_1, x_2, \ldots, x_t)$*, we have:*

$$g(\mathbf{x}) = P(\mathbf{x}) = P(x_t, x_{t-1}, \ldots, x_1)$$

where $P(\mathbf{x})$ *represents the probability of occurrence of the string in some natural language(s).*

Equipped with an appropriate DL network (f_θ), and a dataset $\mathcal{D}$ (*corpus*) containing samples ($\mathbf{x}$) from the languages being modeled, we have:

$$P(\mathbf{x}) = \prod_{i=1}^{t} P(x_i|\mathbf{x}_{<i}) \qquad \mathbf{x}_{<i} = (x_1, \ldots x_{i-1});\ P(x_1|\mathbf{x}_{<1}) := P(x_1)$$
$$\Rightarrow -\log P(\mathbf{x}) = -\sum_{i=1}^{t} \log P(x_i|\mathbf{x}_{<i})$$

Thus, chosing $\mathcal{L}$ such that:

$$\begin{aligned} \mathcal{L}(\theta) &:= -\log P(\mathbf{x}) && \textbf{Negative log-likelihood loss} \\ &\approx \sum_{i=1}^{t} \log[f_\theta(\mathbf{x}_{<i})]_{x_i} \\ \theta^* &= \arg\min_\theta \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}}[\mathcal{L}(\theta)] \end{aligned}$$

gives us a model f_{θ^*}, such that the probability of the corpus is maximised. f_{θ^*} is called an *auto-regressive* or *causal* LM, and

$$[f_{\theta^*}(\mathbf{x})]_i = \text{softmax}\big(g_\theta(\mathbf{x}_{<t}) \cdot e_i\big) \approx P(x_t = i|\mathbf{x}_{<t})$$

where e_i are learned embeddings corresponding to vocabulary token i and g_θ is (typically) a decoder-only transformer model [1, 83, 116].

The other common approach to language modeling [208] is *masked* or *auto-encoder* LM, with the objective:

$$\begin{aligned} \mathcal{L}(\theta) = -\log p_\theta(\tilde{\mathbf{x}}|\hat{\mathbf{x}}) &\approx -\sum_{i=1}^{t} m_i \log p_\theta(x_i|\hat{\mathbf{x}}) \\ &= -\sum_{i=1}^{t} m_i \log\Big(\text{softmax}\big(h_\theta(\mathbf{x}_{<t}) \cdot e_{x_i}\big)\Big) \end{aligned}$$

where h_θ is (typically) an encoder-only transformer model. In this approach, $\hat{\mathbf{x}}$ is a version of a sequence $\mathbf{x}$, corrupted by randomly replacing some ($\sim 10-20\%$) tokens x_j with the `[MASK]` token[6]. The model h_θ then tries to reconstruct $\tilde{\mathbf{x}}$ from $\hat{\mathbf{x}}$. Popular models like BERT [29], RoBERTa [107], etc., follow this paradigm [208].

During training, a technique called *teacher forcing*–where potentially incorrect predictions by the model are used for loss calculation but replaced with the ground truth when calculating representations at other token positions–allows the model to be parallelized.

INFERENCE ALGORITHMS

In order to use a trained *causal* language model for generating text, several schemes like greedy decoding[7], beam search [51], random sampling, etc. have been proposed.

In random sampling, at each step, given a sequence $\mathbf{x}$ of length t, we sample from the distribution $P(x_{t+1} = i|\mathbf{x}) = [f_{\theta^*}(\mathbf{x})]_i$ to get the next token and continue this process until either some predetermined stop condition is reached or $x_T =$ `[EOS]`[8] is predicted. This approach has been found to catalyze diversity in generations, and for more control over diversity we can introduce a scaling parameter T called *temperature* [61]:

$$P(x_t = i|\mathbf{x}_{<t}; T) = \frac{\exp(g_\theta(\mathbf{x}_{<t}) \cdot e_i/T)}{\sum_{j=1}^{V} \exp(g_\theta(\mathbf{x}_{<t}) \cdot e_j/T)}$$

Setting $T = 1$ gives us regular random sampling, and in the $T \to 0$ limit this is the same as greedy decoding. With increasing temperature, diversity increases and yields a max-entropy distribution in the high-temperature limit. Other strategies for controlling generation quality and diversity have been proposed, like Top-k [44], Top-p [73], η-sampling [69], etc., which truncate the distribution following different policies.

PROMPTING

One of the most fortuitous properties of large language models is their *in-context learning* (ICL) ability. Foundational LLMs can perform novel tasks based on provided instructions, a few solved examples, or additional context [15]. This has led to

[6] m_i is an indicator variable, $m_i = 1$ if x_i is masked else 0

[7] Choose the maximum probability next token.

[8] `[EOS]` is a special vocabulary token indicating *end of sequence*.

the thorough exploration of a family of techniques called *prompting*, which attempts to manipulate the input to the LLM, typically the prefix, to improve performance on numerous tasks.

The *few-shot prompting* [15] paradigm involves prepending example inputs and ground truth annotations to the query of an LLM. Brown et al. [15] showed that the addition of a few examples ($\sim 5 - 50$) improves the performance of models across various scales and tasks like translation, reasoning, question answering (QA), etc. Optionally, instructions can be added, which show further promise on *instruction-tuned* LLMs[9] [195]. *Zero-shot prompting* refers to the special case where only instructions or contextual information are added without explicit solved examples.

Chain-of-thought prompting [197] prepends instructions to reason about a problem step-by-step and optionally includes illustrative solutions with logical steps required to arrive at the solution (see Figure 1.2). This approach has been shown to help with reasoning and math tasks [197]. Many such "thought chains" can be sampled and ensembled to improve performance further in an approach called *chain-of-thought* with *self-consistency* [193].

```
Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls.
Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each are 6 tennis balls.
5+6 = 11. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more,
how many apples do they have?

A:
```

FIGURE 1.2: Example of a *chain-of-thought* prompt (reproduced from Wei et al. [197]).

1.3 DOMAIN CONSTRAINTS

As outlined in the preceding sections, DL-based predictive systems outperform other techniques in several areas and also offer unique advantages like being able to handle myriad data types and not being reliant on expert-crafted features. However, a key

[9] Models fine-tuned with instruction following-demonstration datasets.

hindrance to wider adoption is their *unreliability* in certain contexts. In many critical applications, automated decision-making systems *must* exhibit an ability to operate within preset boundaries or *constraints* to be considered for deployment.

It has been demonstrated that conventional DL techniques learning from data alone *do not learn* to operate within the bounds of these requisite constraints [121, 150]. Muralidhar et al. [121] showed that NNs trained with regular approaches disobey monotonicity and boundary constraints, and Saha et al. [150] presented similar findings for logical constraints. The problem is more severe than occasional constraint violation, and Zhang et al. [214] have noted that DL systems can fit "a random labeling of training data"–thus indicating that DL-based approaches are somewhat oblivious to underlying domain principles. This lack of domain awareness is a major deterrent to wider acceptance of DL methods in key domains.

A naive approach to dealing with this issue is to augment DL models with a rule-checking system to suppress offending predictions. Such an approach mitigates this issue somewhat, potentially at the cost of performance, but completely underutilizes the opportunity presented by the constraint information. A constraint-aware system could potentially learn from constraints to make improved predictions, infer missing details, or display enhanced robustness [14, 115]. Further, augmenting such a constraint-aware system (*soft constraint adherence*) with a rule-checker can result in *hard constraint adherence* with minimal performance disruption.

To expound on this, consider the example of a CV system employed to automatically detect credit card details in order to facilitate payment processing. These cards usually feature a checksum scheme [80] to readily detect simple errors in digit entry. A DL-based system that takes into account this constraining information could reinforce its confidence in predicting correct digits or potentially even calculate an occluded digit.

There are two pertinent areas of exploration in this regard: *learning from* and *abiding by* specified domain constraints, which is the focus of this dissertation. A **domain-obedient deep learning** system aims to leverage pre-existing domain constraints in addition to training data to improve prediction performance and better align models to domain expectations.

A related interesting area of study is rule learning or reasoning with DL-based systems [42, 137, 160], where the expected output is novel rules discovered from potentially noisy data. Although there have been some promising results, current

solutions are bottlenecked by their immense computational demands. The constraint adherence problem, although more straightforward, has significant practical ramifications for wider applicability and is the focus of this dissertation. Awareness of domain constraints could also potentially mitigate effects of data sparsity [26, 121].

Constraints can take many forms, like numerical or logical relationships, graphs, probability distributions, or other problem-specific prior knowledge, and we illustrate this with some examples in Table 1.1. Note that domain constraints are a form of domain knowledge; however, all domain knowledge is not necessarily domain constraints; e.g., information regarding the performance of a feature transformation, etc., is domain knowledge but not constraint.

TABLE 1.1: Examples of domain constraints.

Type	**Example**
Numerical	$\forall x,\ f(x) < 5$
	$\forall x, y;\ 7x^2 + 2y = \pi$
	$\forall x_1, x_2;\ x_1 < x_2 \rightarrow f(x_1) < f(x_2)$
Logical	$\forall x, A(x) \rightarrow \neg B(x)$
	$\forall x,$ if $x \in A, x \notin B \rightarrow x \in C$
Graph-based	$\forall x,$ if $x \in \mathbf{Cat} \rightarrow x \notin \mathbf{Reptile}$
	$\forall x, A, B$ if x is of type A and B, x must also be of type ancestor(A, B)
Distribution	$x \sim \mathcal{N}(0, 1)$

The primary focus of this dissertation is on logical constraints. Since graph-based constraints are also typically decomposable as a set of logical constraints, a general framework for incorporating *first-order logic* (FOL) rules into DL systems would address the challenges posed by the former. Dash et al. [26] point out that "Logic is not differentiable", and addressing logical coherence poses a challenge when working in the standard DL framework, which is reliant on gradient-based optimization.

Previous studies in this area have explored techniques like modifying losses, architectures, or transforming datasets [14, 26]. The NN *architecture* employed to address a problem has a strong influence on constraint adherence. For example, consider CNNs, which respect translation equivariance and locality constraints (spatially close pixels are semantically related), or *graph neural networks* (GNNs), which explicitly model node relationships. There have been more explicit capitalizations of this general idea, like the KBANN approach [181], which derives the structure of the NN from domain knowledge expressed as a set of propositional rules. The work by Xie et al. [205] advances a system to incorporate symbolic knowledge expressed as graphs in a GNN to improve generated embeddings. Li and Srikumar [101] proffer adding

connections to the NN based on domain knowledge expressed as FOL rules.

The classical approach to *modifying losses* is to introduce auxiliary objectives penalizing incoherent predictions [112, 121, 166, 206]. Diligenti et al. [32] put forth a system to translate FOL rules to fuzzy constraints, which are then employed as penalty terms. Melacci et al. [115] rephrased domain constraints as polynomials employing continuous logics and transformed the adherence problem into an optimization problem with the polynomials serving as auxiliary losses. Melacci et al. [115] and Sheatsley et al. [165] demonstrated improved adversarial robustness with their techniques. Hu et al. [74] suggested an iterative distillation [71] technique to incorporate logical constraints.

Dataset transformations involve including background knowledge-based relational or logical features extracted from the data alongside the data [48, 97, 207]; however, when considering constraint adherence, the customary approach is to augment the dataset with examples following criteria established by domain rules. As an example, Bjerrum [13] proposes a methodology where they augment the training dataset with synthetic samples that are filtered based on domain constraints to reinforce learning of these constraints. To imbue constraints on input features, data augmentation with constraint-invariant perturbations has also been explored [118, 185].

Despite several promising forays towards logical constraint adherence, a general framework for incorporating logical constraints into DL systems remains elusive [26].

1.4 ORGANIZATION

This dissertation is roughly split into two parts, the *first* of which focuses on an exploration of DL systems' ability to adhere to domain rules. Chapter 2 introduces a new dataset to combat the lack of large-scale rule-annotated datasets and, with its assistance, demonstrates the lack of constraint adherence displayed by SoTA DL-based CV techniques. Chapter 3 analyzes LMs and points out critical deficiencies they exhibit in adhering to domain expectations with the aid of interventions during training and inference. Further, an intervention-based training strategy is proposed that alleviates this effect.

The *second* part of this dissertation introduces new techniques for incorporating

domain constraints into DL-based systems. Chapter 4 puts forth a technique to incorporate logical constraint information into DL systems. This technique leverages domain rules alongside data to disincentivize incoherent predictions and improve predictive performance. Additionally, chapter 4 tackles model evaluation and points out issues with a domain-blind approach to evaluation. A framework for constructing a metric that takes domain knowledge into account is proposed and exemplified with a real-life medical use case. Chapter 5 explores inference with LLMs in a constrained setting. Although significant strides are required in this area, this chapter illustrates how in-context learning paired with a search strategy can enable the application of these models in a constrained setting. This is followed by a few concluding remarks.

2

DO VISION SYSTEMS LEARN RULES?

In this chapter[1], we explore whether DL systems learn to pick up on domain rules when trained on a vast dataset. When analyzing various DL techniques through the lens of domain obedience, it is crucial to have large, high-quality datasets with annotations and constraining rules. Having noticed a gap in this area in the literature, we put forth **VALUED**, or *Vision and Logical Understanding Evaluation Dataset* (Section 2.2). This dataset features 200,000 annotated images of chess games in progress and an associated rule set with the aim of understanding constraint obedience characteristics of SoTA DL techniques. This dataset allows us to explore the compelling question, "*Do models learn rules from data alone?*" (Section 2.3).

We explore the problem of constraint adherence in a *classification* setting, as the vast majority of problems in DL are posed as versions of classification. In addition to regular classification problems, a wide array of tasks like language modeling, semantic segmentation, sentiment analysis, etc., are commonly viewed through the lens of classification. However, MCC, where each sample of data is mapped to exactly one class, does not feature inter-class domain constraints. Thus, we look to a multi-label classification (MLC)-like problem, where many classes may potentially be predicted from a single data instance. This problem poses more complex kinds of errors since the predicted set and ground-truth sets can overlap in diverse ways.

[1]This chapter is largely based on our paper titled *"VALUED - Vision and Logical Understanding Evaluation Dataset"* [150].

2.1 BACKGROUND

Incorporating domain knowledge has been highlighted as one of the "3 Grand Challenges in developing AI systems" by a recent report on AI for Science [171] and has been studied with zeal across several domains. Incorporating domain-specific constraints into DL systems would significantly enhance their application in critical fields, such as robotics, healthcare, law, and material science. Furthermore, the development of these methodologies is expected to diminish the dependency on extensive datasets, which are particularly challenging to annotate in these domains [210]. As DL-based vision systems are increasingly adopted, there is a crucial necessity to assess their logical comprehension and explore methods for integrating such understanding into existing models. A common obstacle identified in the literature is the scarcity of large, high-quality, annotated datasets accompanied by associated rules [121, 204, 210].

To examine the capabilities of vision systems within this scope, it is essential to create a task that ideally possesses the following features:

1. Poses a challenge for SoTA vision foundation models.
2. Contains non-trivial first-order logic rules that can be inferred from data.
3. Presents semantic constraints on localization to test visual reasoning and numerical constraints to test arithmetic reasoning.

Considering these features, we focus on the challenge of recognizing the state of a chessboard from an image of an ongoing game (see Figure 2.1). The objective is to reconstruct the arrangement of pieces on the board. It is important to clarify that our main goal is *not the precise identification of chess game states* [113, 200]. Instead, we aim to *analyze the limitations of current vision systems* in terms of logical comprehension.

BRIEF PRIMER ON CHESS

Chess is a centuries-old two-player board game played on an 8×8 grid, where each player commands 16 pieces of six distinct types. Each type has specific movement rules (legal moves) [46] within the grid. Players alternate turns, moving one piece at a time, aiming to place their opponent in a situation where the king's capture is

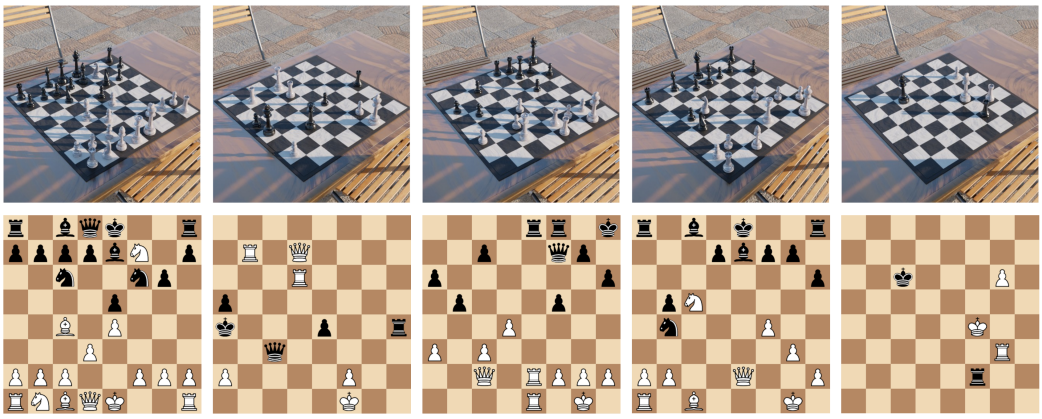


FIGURE 2.1: **Example images from the VALUE Dataset**
The *top row* shows samples from the dataset, and the *bottom row* demonstrates corresponding expected outputs. Figure is reproduced from Saha et al. [150].

inevitable, known as checkmate.

The pieces are differentiated by color—black or white—and are represented by acronyms: `k, q, r, b, n, p` for black king, queen, rook, bishop, knight, and pawn, respectively, and the uppercase equivalents for the white pieces. The chessboard's columns, known as files, are labeled from A to H from left to right, while the rows, or ranks, are numbered 1 to 8 starting from the white side (the bottom leftmost dark square is A1, as depicted in Figure 2.1). The board's entire configuration, or board state, is frequently represented using a shorthand notation known as Forsyth-Edwards Notation (FEN). This format lists the pieces in each rank, separated by slashes ("/"), with numbers indicating consecutive empty squares. For example, in Figure 2.1 the FEN for the leftmost and rightmost boards are `r1bqk2r/ppppbN1p/2n2np1/4p3/2B1P3/3P4/PPP2PPP/RNBQK2R` and `8/8/2k3P1/8/5K2/6R1/5r2/8`, respectively.[2] The game starts from the state `rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR`. We also use $|\mathtt{L}|$ to denote the number of pieces of type `L` on a board (e.g., $|\mathtt{K}| = 1$).

2.1.1. RELATED WORKS

Domain-specific knowledge in the form of discrete logical or numerical constraints is a common element across a variety of problems. Techniques to address this issue often involve modifying the input data [48], adjusting the loss function [206], altering the model architecture [35], or a combination of these strategies [74]. Nonetheless,

[2] `r1bqk2r/...` represents a row with a black rook, followed by an empty square, followed by a black bishop, queen, and king, followed by two empty squares and a black rook arranged from left to right. This is similarly repeated for all ranks (rows).

further research is necessary to establish a standardized framework for embedding domain knowledge constraints into DL systems [26].

A recurrent issue noted in the literature is the scarcity of datasets [35, 204, 210]. Specifically, there is a need for annotated datasets that feature domain-specific constraints to facilitate the evaluation and refinement of constraint obedience methodologies. Often, researchers must rely on toy datasets [137, 160] or those comprising of only a limited number of examples ($\sim 10^3$) [115].

Domain knowledge is often presented as a solution to the challenge of working with sparse datasets, leading to reported results on such datasets [121, 210]. However, when the evaluation of methods is limited to these datasets, the meaningfulness of improvements attributed to domain knowledge remains ambiguous. While, in theory, domain knowledge should help mitigate performance limitations in data-scarce scenarios, this is not an unequivocally resolved issue, and the influence of dataset size on the integration of domain knowledge requires further investigation.

Recently, some advancements have been realized in the NLP domain with the introduction of the BIG-bench (Beyond the Imitation Game benchmark) [170] benchmark, which presents a wide array of logically constrained tasks. However, due to the immense complexity of natural language, constructing a comprehensive set of logical constraints that cover a substantial portion of the domain is challenging. As such, it does not fulfill the requirement for a vast dataset with a comprehensive set of rules that apply to a large portion of domain instances. There are some efforts aimed at evaluating the visual reasoning capabilities of DL systems [5, 84, 213]. However, these datasets focus on the models' proficiency in abstract visual reasoning, and to our knowledge, no high-quality datasets exist for analyzing deductive reasoning based on an extensive set of domain-specific axioms.

Another area where this issue has been extensively explored is in MLC with logical constraints [75, 95, 115]. In these problems, a knowledge graph imposes entailment constraints that predictions must comply with, but typically the set of permissible rules is limited in scope. For instance, datasets like Blurb-genre [4], Uniprot [33], and other works [143, 149, 175], primarily feature rules of the form $\forall x, A(x) \Rightarrow B(x)$. There is a need to explore richer rule sets that incorporate constraints based on factors such as localization and counting, along with basic first-order logic rules. Some large-scale datasets [3, 33] suffer from issues like a long-tailed class distribution and sparse classes, complicating the evaluation of domain-knowledge integration

methods. To further research in this domain, it is vital to develop large datasets with a variety of domain-specific rules, free from extraneous issues like sparsity and class imbalance.

Previous efforts to develop datasets of chess game images [113, 200] were primarily aimed at establishing systems for accurate reconstruction of game states, rather than focusing on the examination of logical understanding. As a result, these datasets do not include a comprehensive rule set or associated metrics necessary for such analysis. Despite containing a comparatively smaller number of images ($\sim 10^4$), these datasets can be subsumed into our framework to enhance diversity and further assist in investigating logical comprehension within vision systems.

In Section 2.2 we discuss the details of our proposed dataset. Finally (Section 2.3), we attempt to analyze the question: *"Given enough data, can deep learning systems trained with standard approaches learn to abide by underlying logic rules?"*

2.2 VALUE DATASET

Implementation Details

To compile the dataset,[3] we initially obtained a massive collection of chess moves from online multiplayer games hosted on Lichess [127], which were then transformed into corresponding intermediate board states by application of the moves to the start position. While this database includes duplicate states, such as those common in opening sequences, we chose to retain them to maintain the original distribution of frequently encountered states.

A 3D model of an environment was created using the open-source modeling software Blender [22]. This environment features a large plane with a table, two chairs, a chessboard, and pieces set in their initial position (see Figure 2.2). Using the board state database, pieces were placed on their designated squares with a small position jitter added to their positions. The scene is rendered from the point of view of a camera aimed at the center of the chessboard and at a fixed distance. Random pans and tilts of the camera were introduced; however, its movement was restricted to ensure that the A1 square remains nearest to it, preventing ambiguity in board orientation.

[3] Dataset - doi.org/10.5281/zenodo.10607059

We rendered 200,000 images at a resolution of $(512 \times 512 \times 3)$ pixels for the training/validation set and an additional 19,967 images for the test set. Besides the board state labels (provided in both array format and FEN), bounding box information for each piece is included to support other methods like object detection, although this data is not used in model evaluation. A concise rule set was established to balance the enforcement of meaningful constraints with computational efficiency. This rule set also facilitates an analysis of the types of errors made by the prediction system, providing insights into how SoTA vision systems operate.

All associated code (database creation, rule checking, etc.), materials (dataset, 3D models, textures, images, etc.), rendering details (camera sensor, rendering settings, etc.), and relevant information have been made available through the github repository.[4]

TASK DESCRIPTION

If this were a standard classification task, we would seek to learn a function $f_\theta : \mathcal{D} \rightarrow [0,1)^{(\#\text{ pieces} + 1)\times 8\times 8}$ (prediction for piece type or empty for all 64 positions) that models the probability of each piece at every board position given an image of the board state. However, when in use, this function must be paired with an inference algorithm with discrete outputs to recreate the board state, which can itself introduce logical inconsistencies. Thus, in order to analyze performance with regard to logical coherence, the predictive model with soft outputs and the inference algorithm are treated together as a black box, and performance is evaluated with respect to the complete board state prediction. Keeping this in mind, we define a classifier as follows:

$$\begin{aligned} F_\theta : \mathcal{D} &\rightarrow \mathrm{P}^{8\times 8} \\ F_\theta(x) &\mapsto \text{board state of } x \end{aligned} \tag{2.1}$$

where $\mathcal{D} \subset \mathbb{R}^{512\times 512\times 3}$ is the space of input images, $\mathrm{P} = \{$`x, p, P, n, N, b, B, r, R, q, Q, k, K`$\}$ is the set of all pieces (`x` represents the empty grid location), and F_θ represents the model we seek, parametrized by θ. The images pose several difficulties from a computer vision perspective, like camera position variability, occlusion (often severe), and dense clusters of similar-looking small objects (see Figure 2.2).

[4] Code repository github.com/espressoVi/VALUE-Dataset

It is important to recognize that this problem could be reformulated as a semantic segmentation problem, a captioning task, or potentially within a completely new machine learning framework, which we do not explore here. This choice is based on the premise that our DL model is envisioned as part of a larger pipeline that performs tasks in real-world applications, where the model's outputs are fed into an automated system. In such applications, demonstrating logical understanding is crucial.

In our example task, if model-generated annotations with minor errors were given to a human, they could easily correct them. However, the impact of logical errors becomes significantly amplified if the model is part of an automated chess-playing robot that also incorporates a chess engine to devise moves and software to manipulate linkages. For this reason, we approach this as a classification problem, recognizing that alternative paradigms would necessitate converting outputs to a standardized format identical to the one specified in Equation 2.1 for subsequent processing.

FIGURE 2.2: **3D environment for synthetic image generation**
The base 3D scene (*left*) and dataset examples demonstrating occlusion (*marked in red*) and object density (*right*). Figure is reproduced from Saha et al. [150].

RULE SET

An arrangement of pieces on the chessboard is called **valid** if there exists a sequence of legal moves [46] from the starting position that results in the arrangement and the set of all valid states is denoted $\mathcal{V}$. Given the state of a chessboard, it is computationally prohibitive to determine if the state is valid; however, some simple *sanity checks* can be utilized to rule out the vast majority of invalid states. We curated such a set of computationally cheap ($\mathcal{O}(nd^2)$ for n chessboards of side d, i.e., $\mathcal{O}(n)$) first-order logic rules that hold true for all valid chessboard states, to measure domain

coherence. The proposed rule set is given in Table 2.1 (equivalent rules for white pieces are also included).

TABLE 2.1: **Rule set associated with VALUE Dataset.**
A valid chess state must obey all rules (i-viii) given below. These rules are further divided into two categories–**counting** (i, iii, iv, vi, vii), and **localizing** (ii, v, viii), to analyze specific semantic abilities.

	$\forall y \in \mathcal{V}$ (valid states), we have:	
i	$\|\mathsf{k}\| = 1$	Exactly one king.
ii	k, K are not on adjacent squares.	
iii	$\|\mathsf{p}\| + \|\mathsf{q}\| + \|\mathsf{n}\| + \|\mathsf{b}\| + \|\mathsf{r}\| \leq 15$	Total number of pieces, including king cannot exceed 16.
iv	$\|\mathsf{p}\| \leq 8$	Total number of pawns not exceeding 8.
v	$\forall \mathsf{p}, 2 \leq rank(\mathsf{p}) \leq 7$	No pawn on first or last rank.
vi	$(\|\mathsf{p}\| = 8) \Rightarrow (\|\mathsf{q}\| \leq 1) \wedge (\|\mathsf{b}\| \leq 2) \wedge (\|\mathsf{n}\| \leq 2) \wedge (\|\mathsf{r}\| \leq 2)$	If no pawn is promoted, there cannot be more than two bishops, knights, rooks or more than one queen.
vii	$(\|\mathsf{p}\| < 8) \Rightarrow \max(0, \|\mathsf{q}\| - 1) + \max(0, \|\mathsf{b}\| - 2) + \max(0, \|\mathsf{n}\| - 2) + \max(0, \|\mathsf{r}\| - 2) \leq 8 - \|\mathsf{p}\|$	If pawns might have been promoted, the number of excess pieces cannot exceed the number of missing pawns.
viii	$(\|\mathsf{p}\| = 8) \wedge (\|\mathsf{b}\| = 2) \Rightarrow \mathsf{b}_1, \mathsf{b}_2$ don't occupy squares of the same color.	If no pawn has been promoted and there are two bishops, they must be on opposite color squares.

If a prediction y *satisfies all the rules* (Table 2.1), we call it **sane**, and we have the *set of sane states* $\mathcal{S}$ following $\mathcal{V} \subsetneq \mathcal{S} \subsetneq \mathrm{P}^{64}$. These rules (Table 2.1) are further divided into two categories–**counting** (i, iii, iv, vi, vii), and **localizing** (ii, v, viii), to analyze specific semantic abilities. Counting rules apply constraints on the number of objects that can be present in the scene, whereas localizing rules apply constraints on their position in the scene.

This is not an exhaustive list, and more such rules can be found. Consider the following examples:

- b can't be trapped in the last rank behind 3 adjacent P.
- If all pawns are in their starting position, only the knights can occupy a rank greater than 2.
- Both kings cannot simultaneously be in check.
- If there are multiple pawns on the same file, the total number of these excess pawns per file cannot exceed the number of the opponent's missing pieces (as pawns can only change files through capture).

To the best of our knowledge, we have included all rules that are:

- Generalizable to real-world scenarios, such as counting and relative position constraints.

- Computationally cheap to verify.
- Not reliant on infrequent game states, i.e., they occur frequently enough in the training set for vision models to potentially learn them.

Adding more niche rules could make the analysis overly specific to chess, which would detract from our broader goal of examining the general capability of vision models to learn logical constraints. Note that the general problem, i.e., determining whether a sequence of legal chess moves can lead to a particular board state, is computationally intractable. Moreover, we do not expect vision models trained solely on images of board states to acquire this information. For instance, consider the following rule: If k or K is in check (can be captured in the next move), then there must have existed at least one legal move for the piece threatening capture that results in the current state. To learn such rules, the model would need to understand legal piece movements and the rules for check, which cannot be inferred from state images alone.

These rules are nonetheless effective at eliminating a large fraction of invalid states. Note that the total number of distinct predictions we would have following standard DL approaches (64 independent classification problems) is $|\mathrm{P}^{64}| = 13^{8\times 8} \sim 10^{72}$, but with the addition of just the constraint on the number of pieces, it reduces to $< 10^{55}$.

If these simple rules can be incorporated into the learning algorithm, in addition to being more applicable to the domain, it could in principle improve performance drastically. For example, if q was misclassified as k, or b was misclassified as p, resulting in $|\mathsf{p}| = 9$, the rule set would identify and seek to disincentivize it.

EVALUATION METRICS

We employ several standard metrics to assess both raw performance and alignment with domain constraints. Given that the logical constraints are applied to the discretized, complete board state prediction, and in accordance with our definition in Equation 2.1, techniques such as "threshold tuning" or similar inference methods are integrated as part of the model evaluation process. Hence, commonly used metrics like Area Under the Precision-Recall Curve (AUPRC), Area Under the Receiver Operating Characteristic Curve (AUROC), or macro-averaged metrics such as mean Average Precision (mAP) are not included in our analysis. Given a prediction set $\hat{Y} = \{\hat{y}_i | i \in \{1, \ldots, n\}\}$ and the corresponding ground truth set $Y = \{y_i\}$, where

$y_i, \hat{y}_i \in \mathrm{P}^{64}$, we define exact match (EM) and F1 as follows:

$$EM(Y, \hat{Y}) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}([y_i]_k = [\hat{y}_i]_k),\ \forall k, 1 \leq k \leq 64) \tag{2.2}$$

$$\begin{aligned} F1(Y, \hat{Y}) &= \frac{2}{n} \sum_{i=1}^{n} f_1(y_i, \hat{y}_i), \text{ where} \\ f_1(y_i, \hat{y}_i) &= \frac{\sum_{j=1}^{64} \mathbb{I}([y_i]_j = [\hat{y}_i]_j \neq \mathsf{x})}{|y_i| + |\hat{y}_i|} \end{aligned} \tag{2.3}$$

Where $|y| = \sum_j \mathbb{I}([y]_j \neq \mathsf{x})$ denotes the number of non-empty squares in the grid, and $\mathbb{I}$ is the indicator function that takes the value 1 if its argument condition is true, and 0 otherwise.

Additionally, we define two measures of domain coherence–**contradiction %** (C) and **sane F1** $(sF1)$–as follows:

$$C(Y) = \frac{100}{n} \sum_{i=1}^{n} \mathbb{I}(\hat{y}_i \notin \mathcal{S}) \tag{2.4}$$

$$sF1(Y, \hat{Y}) = \frac{2}{n} \sum_{i=1}^{n} \Big(\mathbb{I}(\hat{y}_i \in \mathcal{S}) \cdot f_1(y_i, \hat{y}_i) \Big) \tag{2.5}$$

C reflects the frequency of logical constraint violations, i.e., what fraction of predictions are unusable, and the $sF1$ score measures the F_1 score after eliminating predictions that are not sane. We also report results on μ_C, the mean number of rule violations per instance in $\hat{Y}$.

$$\mu_C(\hat{Y}) = \frac{1}{n} \sum_{i=1}^{n} \Big(\text{\# of rule violations in } \hat{y}_i \Big) \tag{2.6}$$

When integrating the model into an automated system with downstream tasks requiring strict compliance with domain rules, it is crucial to suppress predictions that violate these guidelines by implementing consistency checks against the established rule set. The $sF1$ measure is introduced to account for this counterfactual scenario, where predictions that breach the rules are substituted with a null prediction set. The gap between the traditional $F1$ score and the $sF1$ score indicates the potential for improvement available to algorithms that aim to incorporate logical constraints effectively. This distinction highlights the impact of domain-specific knowledge on enhancing the model's reliability in practical applications.

To summarize, in this task, we seek an F_θ, such that given a set of images $x \in \mathcal{D}$, it can faithfully recreate corresponding ground truth labels $y \in \mathrm{P}^{64}$ while minimizing violations of domain rules, i.e., $sF1\Big(\{F_\theta(x_i)\}, \{y_i\}\Big)$ is maximized.

2.3 ARE RULES LEARNT?

Given our dataset of 200,000+ annotated chess games, we can test the capabilities of various vision models with regard to their performance and domain constraint adherence.

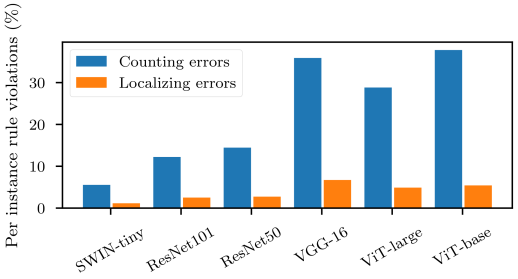
To establish baseline results, we selected a range of popular ImageNet [28] pre-trained vision models like ResNets [65], ViT [34], etc., covering a large range of scales and training techniques. These pre-trained models were fine-tuned [5] for 2 epochs after replacing the final FC layer with one of appropriate size ($\texttt{in\_features} \rightarrow 8 \times 8 \times \texttt{class\_number}$) and adding dropouts (10%) in between. The models were implemented in pytorch, and trained with the AdamW optimizer (learning rate of 10^{-4}) and CE loss. The images were normalized, and if necessary resized. We trained the models on a single NVIDIA A6000 48GB GPU.

2.3.1. RESULTS

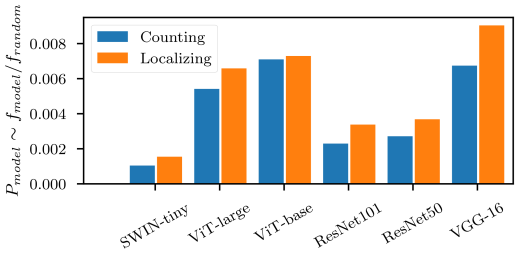
The performance analysis of various vision models on our dataset is presented in Table 2.2. Although these pre-trained models differ in terms of training techniques and scale, they demonstrate proficiency in recognizing visual features necessary for identifying chess pieces, as evidenced by their high $F1$ and EM scores (see Table 2.2). Interestingly, the Swin Transformer [108] surpasses models that are significantly larger, potentially due to its unique hierarchical architecture that allows for effective feature capture at varying scales.

However, these models show considerable room for improvement in terms of domain consistency, as indicated by the high percentage of predictions containing rule violations. In critical applications, such inconsistent predictions are untenable and would require elimination, which is reflected in the notably lower $sF1$ scores. To underscore this issue, we draw attention to the μ_C metric in Table 2.2, which records the average number of rule violations per prediction. In the best-case scenario, a rule violation occurs every 15 predictions, while in the worst case, it occurs every 2.3

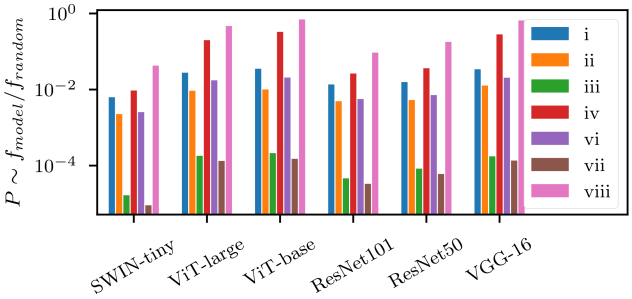
[5] Full fine-tune updating all weights.



(a) Prevalence of violating counting or localizing rules.



(b) Adjust likelihood of violating counting or localizing rules (Equation 2.7).



(c) Adjusted likelihood of violating each rule.

FIGURE 2.3: Errors arising due to counting or localizing.

predictions.

Although the results discussed pertain to off-the-shelf vision systems, it is worth noting that the performance of systems specifically designed for the chess board state recognition task is comparable, with EM scores of approximately 15% and 7% for end-to-end and piece-wise classification systems, respectively [113]. This highlights the ongoing challenge of reducing logical inconsistencies in model predictions.

TABLE 2.2: **Performance of popular vision models on the VALUE dataset.** ([↑] - higher is better, [↓] lower is better).

Model	# param.	Image Size	EM (%)[↑]	C (%) [↓]	F1 [↑]	sF1 [↑]	F1-sF1 [↓]	μ_c[↓]
VGG-16 [167]	134M	224^2	26.3%	28.7%	0.880	0.656	**0.224**	0.426
ResNet50 [65]	24M	512^2	56.3%	12.8%	0.959	0.849	**0.110**	0.172
ResNet101	43M	512^2	60.4%	11.1%	0.966	0.869	**0.097**	0.147
ViT-B/16 [34]	86M	224^2	25.9%	30.8%	0.875	0.635	**0.240**	0.432
ViT-L/32	307M	384^2	32.2%	24.0%	0.907	0.711	**0.196**	0.337
SWIN-tiny/4 [108]	29M	224^2	80.3%	5.2%	0.984	0.938	**0.046**	0.067

Given that our rule set includes several rules specifically tailored to evaluate localization ability, i.e., constraints on where objects can be located in the scene, and counting ability, i.e., constraints on the number of objects of a certain kind in the scene, we examined how well models perform along these categories. Although

ostensibly it is evident that most errors made by the models stem from breaching counting constraints (see Figure 2.3a), it is important to recognize that the probability of making counting errors is elevated due to the significantly greater number of potential predictions that can infringe upon these constraints. To adjust for this, we calculate-

$$P^{\mathrm{R}}_{model} = f^{\mathrm{R}}_{model} / f^{\mathrm{R}}_{random} \tag{2.7}$$

where f^{R}_{model} is the frequency of rule R being violated by the model in question, and f^{R}_{random} is the frequency of rule violation of a random guesser ($P(s) = P(s'), \forall s, s' \in \mathrm{P}^{64}$). This shows (see Figure 2.3b) that the models are more likely to make errors in regard to localization ($\mu = 0.0052 \pm 0.0025$) than counting ($\mu = 0.0042 \pm 0.0023$).

Additionally, we examined the models to assess the likelihood of each rule being violated within the two main categories. Surprisingly, rule (v) in Table 2.1 was never violated by any model, while rules (iii) and (vii) were unlikely to be violated (see Figure 2.3c). Conversely, locality constraints (ii) and (viii), alongside counting rules (i), (iv), and (vi), were frequently violated. This points to the fact that the ability of models to adhere to domain constraints is somewhat dependent on the nature of the constraints themselves.

2.3.2. DISCUSSIONS

Our study reveals that while conventional DL methods may appear effective when evaluated using standard metrics on the VALUE Dataset (refer to Table 2.2), they often struggle to comply with domain-specific constraints. Even among the highest-performing models, 5.2% of predictions (up to 30% in the worst case) exhibit logical inconsistencies, which are unacceptable in critical applications and result in a reduced effective F1 score (sF1). Additionally, we found that these DL approaches exhibit a limited ability to learn constraint-related information from data alone, a capability that significantly varies based on the type of constraint. Moreover, it remains uncertain whether the enhanced performance of certain models (as shown in Table 2.2) is due to their ability to align with dataset constraints or if such results are meretricious. Further exploration into model robustness is also necessary, particularly since integrating domain knowledge has shown potential in enhancing adversarial robustness [115].

Given the significant practical relevance of the problem of incorporating domain knowledge into DL systems, this subject has garnered considerable attention. How-

ever, the scarcity of high-quality datasets featuring a diverse and well-curated set of rules has limited thorough analyses and the advancement of incorporation techniques. The VALUE Dataset represents a step toward addressing this gap. Nonetheless, additional research is essential to establish a standardized framework for integrating domain constraints into deep learning methodologies. Further investigations into how different characteristics of these rules—such as compositionality, specificity, and their prevalence in the training distribution—challenge SoTA vision systems constitute a promising avenue for future work.

3

FAITHFUL LANGUAGE MODELING

[1]Despite transformer-based LMs showing remarkable performance at a multitude of NLP tasks, their internal consistency leaves a lot to be desired. In this chapter we analyze a set of popular transformer-based LMs to gauge their ability to adhere to the semantics of the provided context. In particular, we explore contextual QA tasks, wherein some contextual information *(story)* is provided to the LM, and a question is posed whose answer is present in the supplied context. We argue that domain-aligned LMs should be robust to manipulations of the provided contextual information, and their responses should change if key pieces of provided context are altered. This notion is termed *faithfulness*.

We develop a simple and cost-effective intervention strategy called *deletion intervention* that can manipulate key information in the provided context and apply it to a range of LMs. Our findings show that these models are unable to maintain consistency in this scenario. We further propose an intervention-based training (IBT) technique that can mitigate this issue and better align models to domain expectations.

In the following sections we outline some necessary background, followed by our experimental results demonstrating a lack of semantic faithfulness. We also present

[1]This chapter is largely based on our paper titled *"Analyzing Semantic Faithfulness of Language Models via Input Intervention on Question Answering"* [18]. Chaturvedi et al. [18] is much wider in scope and establishes a formal notion of semantic faithfulness; we restrict ourselves to discussions of adherence of domain constraints *vis à vis* QA.

results of further analyses, analyzing the efficacy of the IBT scheme with regard to imbuing semantic faithfulness in LMs.

3.1 BACKGROUND

A considerable body of research has explored how language models perform across various NLP tasks [139]. One such prevalent method is *probing*. This technique is employed to uncover linguistic structures present in the contextual vector embeddings of these models [20, 68, 70, 133]. The process of probing involves training a supplementary model called the *probe*, which utilizes the representations obtained from an LM for a downstream linguistic task. Since the LM representations are fixed, the performance of the probe model indicates how well these representations embed information necessary for the downstream task. The simplest example of this technique is *linear probing* [2], which tests the linear separability of LM-generated features with an FC layer.

A significant limitation of probing methods is their inability to elucidate how the embedded information is utilized during the inference process [139, 178]. Probing only examines the presence of sufficient information within the representation but tells us nothing about whether the model actually employs this implicit information when reasoning about textual content. The focus of our experiments is squarely on this aspect, i.e., how the model makes use of the embedded information in its reasoning process.

Methods that study LM behavior at inference time can offer better insights into the inner machinations of LMs. One such work by Elazar et al. [38] explores an intervention-based strategy called *amnesic probing*. This method involves making alterations to the hidden representations of the model to selectively erode certain morphological information. Our work, in contrast, focuses on performing interventions on the input linguistic content and form. In related research, Balasubramanian et al. [9] demonstrated that BERT can be *surprisingly brittle* when swapping one named entity for another. Furthermore, Sun et al. [173] highlighted BERT's lack of robustness to common typographical errors.

Schuff et al. [154] explored QA models that provide explanations alongside answers. Their manual inspection revealed that the explanations often failed to justify the predicted answers. Studies investigating the effect of manipulated input texts [9,

11, 81, 169, 173, 216] typically do so through the lens of *adversarial* scenarios, employing attack algorithms or intricate heuristics to alter outputs, even when they shouldn't change. Interventions serve as a crucial tool in crafting counterfactual models [89], and they provide insights into understanding causal structure [10, 98, 155]. Elazar et al. [37] introduced a notion of consistency, which is subsumed by the notion of semantic *faithfulness* in our work [18].

3.2 METHODOLOGY

DATASETS AND TASK DESCRIPTION

Our experiments are performed on the *CoQA* [136] and the *HotpotQA* [209] datasets. CoQA stands for Conversational Question Answering and features story passages *(context)* from several domains like children's stories, news, etc., alongside multi-turn conversational questions. Additionally, a span of the paragraph containing key information required to answer (*rationale*) the question is annotated. The answers are typically also a span of the story or are one of {yes, no, 0∼9, unknown}. The various questions associated with each story might be correlated, i.e., a follow-up question based on previous questions or answers is featured. An example instance of the CoQA dataset can be found in Table 3.1.

TABLE 3.1: **Example question from the CoQA dataset.**
The text marked in **bold** is the associated annotated rationale required to answer the question.

Story	Once upon a time, in a barn near a farmhouse, there lived a little white kitten named Cotton. Cotton lived high up . . . farmer's horses slept. **But Cotton wasn't alone in her little home above the barn, oh no.**
Conversation History	What color was Cotton? white Where did she live? in a barn
Question	Did she live alone?
Answer	no

HotpotQA is a single-turn QA dataset, and corresponding to each question there are 2 gold and 8 distractor context paragraphs sourced from *Wikipedia*, with only the gold paragraphs containing information relevant to answering the query. This dataset also contains *rationale* annotations, and for parity with the CoQA dataset, we use

the two concatenated gold paragraphs as the provided context *(story)*.[2] An example instance of the HotpotQA dataset can be found in Table 3.2. The dataset statistics are provided in Table 3.3, and results are reported on the *dev* sets of these datasets.

TABLE 3.2: **Example question from the HotpotQA dataset.**
The text marked in **bold** is the associated annotated rationale required to answer the question.

Story	**Chumbawamba were a British rock band that formed in 1982 and had major success until their final performances in 2012.** The band constantly shifted in musical style, drawing on . . . **Spin Doctors is a rock band from USA, formed in New York City**, best known for their early 1990s hits, "Two Princes" and "Little Miss Can't Be Wrong", . . .
Question	Are Chumbawamba and Spin Doctors from the same country?
Answer	no

TABLE 3.3: Data Statistics for CoQA and HotpotQA along with the percentage of *unknown* questions.

Dataset	Split	Story	Questions	unk%
CoQA	train	7,199	108,647	1.26
	dev	500	7983	0.83
HotpotQA	train	84579	90447	-
	dev	7350	7405	-

MODELS AND EXPERIMENTAL DETAILS

We employed widespread LMs like BERT [29], RoBERTa [107], and XLNet [208] in their *base* and *large* variants. Additionally, InstructGPT [128], i.e., `text-davinci-002` and `text-davinci-003`, were used for this study. The architectures were unchanged between the two datasets. The input was provided in the following format:

$$\Big[[Q^{i-2}_{1...k_1}], [A^{i-2}_{1...k_2}], [Q^{i-1}_{1...k_3}], [A^{i-1}_{1...k_4}], [\mathbf{Q^i}_{1...k}], [SEP], [S_{1...n}]\Big]$$

where $[Q^m_{1...k}], [A^m_{1...k}], [S_{1...k}]$ refer to the k tokens of the m^{th} turn question, answer,[3] and the story, respectively.[4] Additional tokens like $[CLS]$, etc., were added as needed. The publicly available XLNet implementation for CoQA[5] was used, and BERT and RoBERTa were implemented according to the work by Ju et al. [85]. We implemented

[2]For the purposes of this study, the distractor paragraphs were discarded.
[3]Omitted for HotpotQA.
[4]Context is placed before question for XLNet.
[5]github.com/stevezheng23/mrc_tf

TABLE 3.4: **Example of information repetition in the CoQA dataset.**
Just removing the *rationale* from the story is often not enough to remove critical information. However, since the first instance of the critical information is always annotated, upon truncation, it should not be possible for LMs to respond to the query. The annotated *rationale* is marked in **bold**, and repetitions are shaded.

Story	Characters: Sandy, Rose, Jane, Justin, Mrs. Lin . . . Jane: Sandy, I called you yesterday. Your mother told me . . . This year is very important to us. **Sandy: (crying) My father has lost his job**, and we have no money to pay all the . . . . . . Jane: Eh... I hear that Sandy's father has lost his job . . .
Question	Who was unemployed?
Answer	Sandy's father

the local LMs in PyTorch [130] using the Huggingface library [199]. The code and other necessary materials to reproduce our results have been open-sourced.[6]

Our proposed method for analysis, called *deletion intervention*, removes the *rationale*, i.e., necessary information from the context required to answer a query. In some instances of the CoQA dataset, there is some information redundancy in the form of repetitions of necessary information; however, in all such instances the first occurrence of the information is annotated as the *rationale* (see Table 3.4 for an example). Armed with this information, we perform the following interventions and create corresponding datasets:

- **OS** (*Original Story*) - The original dataset without any changes.
- **TS** (*Truncated Story*) - Dataset with the stories truncated after, but including, the rationale.
- **TS-R** (*Truncated Story - Rationale*) - Dataset with the stories truncated just before the rationale, with the ground truth answer appended to the end.
- **TS-R+Aug** (*Truncated Story - Rationale + Augmentation*) - Same as **TS-R**, but instead of the ground truth answer, a synthetic sentence containing the ground truth is appended.

The **OS** and **TS** interventions contain enough information to successfully respond to the query. The **TS-R** intervention, however, removes this requisite information, only retaining the ground truth. For example, consider the query, *"Where does David go after work?"* and the story–*"David works in an office. He goes to the gym after work."* . The **TS-R** intervention would result in the story–"David works in an office. gym". To study if superficial cues, introduced by the intervention process **TS-R**,

[6] github.com/akshay107/sem-faithfulness

are relied upon by the models, we create **TS-R+Aug**, which has the ground truth information phrased in the form of a sentence with the help of `gpt-3.5-turbo`. Under **TS-R+Aug**, the previous exemplar story would be, *"David works in an office. Going to the gym helps you stay fit."*. Since the **TS-R** and **TS-R+Aug** datasets do not contain enough information to answer the question, a *faithful* LM should answer queries from these datasets with *unknown*.

The HotpotQA dataset does not feature *rationale* repetition, and thus we do not perform truncation on it. The three intervention schemes on this dataset are **OS**, which maintains the original story; **OS-R**, which has the rationale removed and the ground truth appended; and **OS-R+Aug**, which has the rationale removed and the ground truth appended in the form of a synthetic sentence.

INTERVENTION-BASED TRAINING

For evaluating the open-source LMs, we performed fine-tuning. Following conventional practice, the models were fine-tuned on the unmodified dataset, **OS**,[7] and this strategy is referred to as **OT** for *original training*. They were then evaluated under the various interventions outlined in the preceding section. Additionally, we propose a new *intervention-based training* **(IBT)** strategy. This training strategy involves sampling from the combined **OS, TS**, and **TS-R** (**OS**, and **OS-R** for HotpotQA) intervened CoQA datasets and changing the ground truth answers to reflect faithfulness. This entails preserving the ground truth answers for the **OS** and **TS** splits and modifying the ground-truth answers to *"unknown"* for the **TS-R** and **OS-R** splits on the CoQA and HotpotQA datasets, respectively.[8] In the following section we present empirical evidence demonstrating that **IBT** can mitigate the unfaithfulness demonstrated by LMs'.

3.3 EXPERIMENTS

3.3.1. FAITHFULNESS

The results of our experiments are summarized in Table 3.5 and Table 3.6. We first note that there is no appreciable performance difference between the **OS**- and **TS**-intervened CoQA datasets, which is in line with expectations. Further, our

[7] For 1 epoch.
[8] The augmented datasets **TS-R+Aug** and **OS-R+Aug** are only used for evaluation.

TABLE 3.5: **Performance of the models on the CoQA dataset.**
$unk\%$ refers to the percentage of answers predicted as unknown by the models. Since TS-R and TS-R+Aug have critical context removed, we expect EM and F1 to decrease and $unk\%$ to increase. Models trained in the regular scheme (**OT**) do not follow this, but models trained with **IBT** do.

Model	Dataset	OT			IBT		
		F1	EM	$unk\%$	F1	EM	$unk\%$
BERT-base	OS	76.1	66.3	1.97	76.4	67.2	3.82
	TS	77.2	67.1	2.18	77.7	68.0	7.93
	TS-R	55.6	48.2	1.98	5.7	5.4	93.08
	TS-R+Aug	51.5	44.3	7.10	42.4	36.3	45.50
BERT-large	OS	80.7	71.1	2.01	78.8	69.8	4.20
	TS	81.6	72.1	2.32	80.1	70.7	7.34
	TS-R	63.6	57.8	3.79	5.4	5.1	94.25
	TS-R+Aug	53.4	46.3	8.80	38.3	32.9	51.88
RoBERTa-base	OS	80.3	70.8	1.95	81.2	71.6	2.86
	TS	80.8	71.1	2.64	81.9	72.0	5.20
	TS-R	55.5	51.1	16.92	5.5	5.3	94.25
	TS-R+Aug	39.2	28.2	14.26	6.3	6.0	92.13
RoBERTa-large	OS	87.0	77.7	1.74	86.2	76.9	2.66
	TS	86.8	77.3	2.72	86.3	76.7	4.01
	TS-R	59.9	55.7	22.36	5.1	5.0	95.34
	TS-R+Aug	42.9	32.0	22.18	6.0	5.7	93.65
XLNet-base	OS	82.5	74.8	1.08	81.3	74.2	4.63
	TS	82.1	74.2	1.11	79.6	72.4	10.87
	TS-R	53.5	48.0	14.00	6.6	6.4	93.86
	TS-R+Aug	50.7	45.3	13.45	25.2	22.2	68.75
XLNet-large	OS	86.3	78.9	0.86	83.1	75.8	5.10
	TS	85.6	78.5	2.58	81.0	74.1	10.69
	TS-R	48.1	44.3	31.68	5.6	5.5	95.42
	TS-R+Aug	46.9	42.4	26.18	22.1	19.5	73.93

experiments indicate that all tested models demonstrate poor semantic *faithfulness* on both the CoQA and HotpotQA datasets.

Given that **TS-R, TS-R+Aug, OS-R**, and **OS-R+Aug** do not contain critical information required to answer queries, we expect that their performance on these would be near zero as measured by EM and F1. Additionally, on these splits, we should also expect $unk\%$–which measures the percentage of queries with the *unknown* response– to be very high. For models trained with the **OT** strategy, we find that this is not the case. Although there is some dip in performance on the ***-R** and ***-R+Aug** sets, the models do not *change their answers despite critical information being deleted* in the vast majority of cases. The situation is worse on the HotpotQA dataset, and the RoBERTa model even manages to provide answers more accurately with critical information removed, as evidenced by the higher EM% on **OS-R** compared to **OS**,

TABLE 3.6: **Performance of the models on the HotpotQA dataset.** $unk\%$ refers to the percentage of answers predicted as unknown by the models. Since OS-R and OS-R+Aug have critical context removed, we expect EM and F1 to decrease and $unk\%$ to increase. Models trained in the regular scheme (**OT**) do not follow this, but models trained with **IBT** do.

Model	Dataset	OT			IBT		
		F1	EM	$unk\%$	F1	EM	$unk\%$
BERT-base	OS	72.3	56.7	0.16	71.2	55.8	1.00
	OS-R	59.5	48.5	4.60	0.4	0.3	99.05
	OS-R+Aug	63.1	52.2	4.27	3.1	2.2	93.96
BERT-large	OS	74.8	59.6	0.21	73.8	58.5	1.24
	OS-R	63.7	53.8	5.63	0.5	0.4	99.17
	OS-R+Aug	64.7	54.2	5.36	2.63	2.05	95.46
RoBERTa-base	OS	72.0	56.7	0.16	72.7	57.4	0.73
	OS-R	66.2	59.1	0.86	0.6	0.5	98.86
	OS-R+Aug	36.9	15.7	0.94	0.9	0.6	97.93
RoBERTa-large	OS	80.0	64.5	0.18	79.7	64.4	0.70
	OS-R	75.2	70.0	2.84	0.6	0.5	99.06
	OS-R+Aug	40.3	18.4	3.90	0.9	0.6	97.86
XLNet-base	OS	74.2	60.1	0.07	73.5	59.4	1.05
	OS-R	63.0	53.0	0.73	0.6	0.4	98.85
	OS-R+Aug	63.5	53.9	1.16	3.1	2.4	94.30
XLNet-large	OS	80.0	66.1	0.23	77.4	63.5	1.03
	OS-R	68.5	59.1	0.60	0.4	0.3	99.21
	OS-R+Aug	62.7	53.7	9.12	1.6	1.1	96.81

which is contrary to domain expectations. The fact that the $unk\%$ does not change significantly throughout the interventions lends further credence to the hypothesis that these models are not semantically faithful.

Models trained with the **IBT** strategy buck this trend. We observe that although performance on the **OS** set is nearly identical to the models trained with the **OT** strategy, their performance on the ***-R** and ***-R+Aug** sets is much worse. Since performance is measured with respect to the unaltered original ground truth, i.e., ground truth corresponding to **OS**, this is the desired behavior for semantically faithful models. Additionally, almost all queries in these sets are marked as *unknown*, which is the *faithful* answer.

Our experiments with InstructGPT[9] [128] also revealed a troubling lack of *faithfulness*. We present performance results of the models (see Table 3.7) as measured by EM and F1 scores for the two models on **TS-R** and **TS-R+Aug** datasets for CoQA and **OS-R** and **OS-R+Aug** datasets for HotpotQA. A basic prompt consisting of only

[9] https://platform.openai.com/docs/models/gpt-3-5

TABLE 3.7: **Faithfulness performance of InstructGPT models.**
Deletion Intervention: EM and F1 scores for the two InstructGPT models.

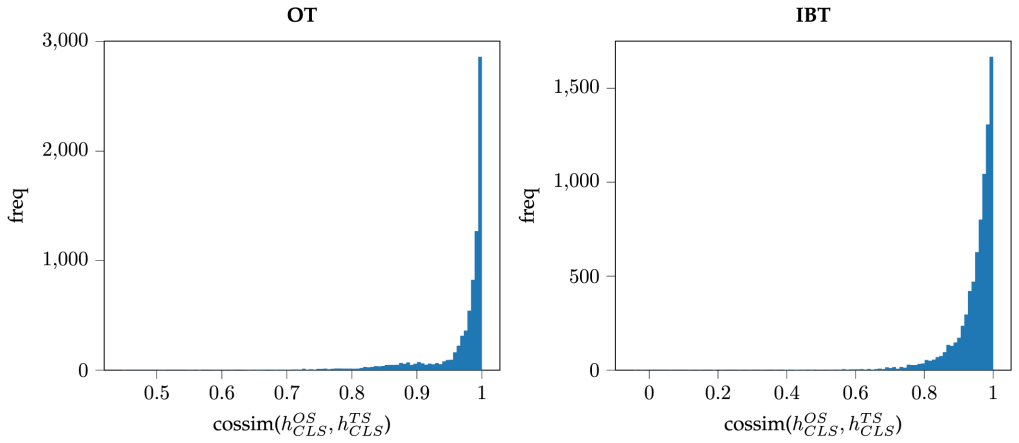
Model	Dataset		EM	F1
text-davinci-002	CoQA	TS-R	46.4	58.5
		TS-R+Aug	40.4	53.3
	HotpotQA	OS-R	14.1	32.2
		OS-R+Aug	11.4	29.5
text-davinci-003	CoQA	TS-R	28.0	45.6
		TS-R+Aug	17.3	34.9
	HotpotQA	OS-R	25.6	41.7
		OS-R+Aug	18.0	34.1

the story and the question was provided. For CoQA, we see that *text-davinci-002* receives astounding scores and correctly predicts the ground truth for nearly $\sim 50\%$ of samples despite the removal of relevant information. While *text-davinci-003*'s scores are relatively modest, it still fabricates the ground truth answer for $\sim 30\%$ of instances. Unlike CoQA, for HotpotQA, *text-davinci-002* achieves lower scores than *text-davinci-003* on **OS-R**. Similar trends are also noted on the HotpotQA dataset, with *text-davinci-003* predicting the ground truth answer for 25.6% of instances of **OS-R**.

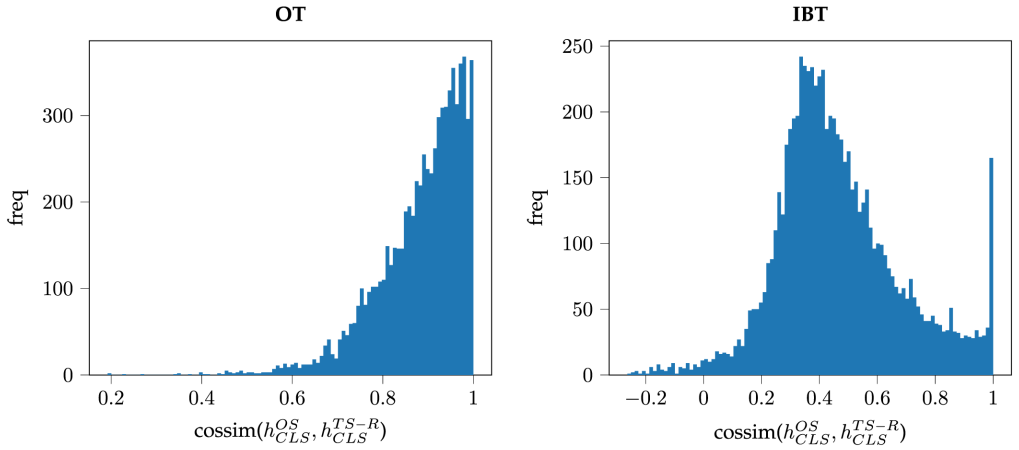
Notably, the scores for both models are lower for **TS-R+Aug** and **OS-R+Aug** than for **TS-R** and **OS-R** on the CoQA and HotpotQA datasets, respectively. This result, alongside the overall poor semantic faithfulness, suggests that the models rely on superficial clues to arrive at an answer. For example, given a query like *"What color is X?"* the models look for any words that fall in the color category to respond with instead of focusing on the semantics of the context.

3.3.2. UNDERSTANDING IBT

We perform further experiments to analyze the efficacy of the **IBT** training strategy with deletion intervention. To this end, we employ the cosine similarity ($cossim$) measure on the output embeddings produced by the models on the various intervention-based datasets. The $cossim$ of output embeddings is an informative measure since embeddings with high $cossim$ are extremely likely to be treated similarly by the downstream FC layer, thus leading to similar final model predictions. We first analyze this with the final layer $[CLS]$ embeddings, as they represent summarized contextual information necessary for classification. We measure the $cossim$ between embeddings produced by the model on the **TS** and **TS-R** datasets against the unmod-



(a) **OS** and **TS**

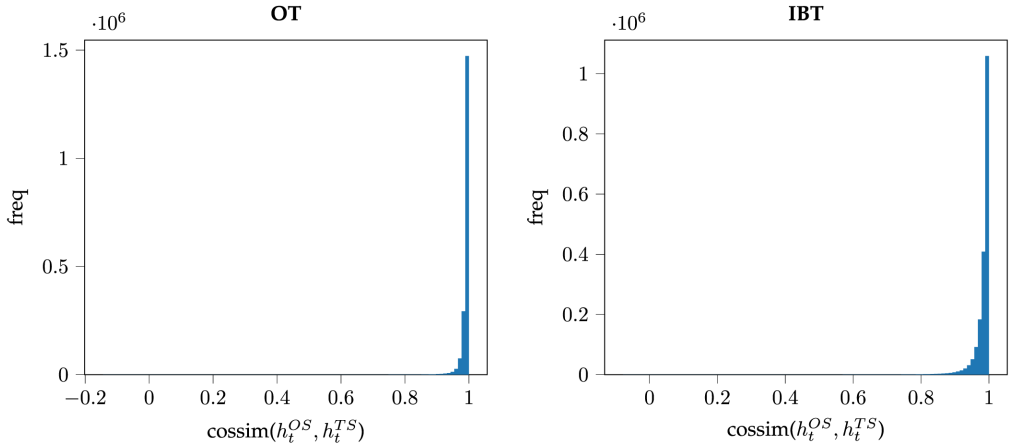


(b) **OS** and **TS-R**

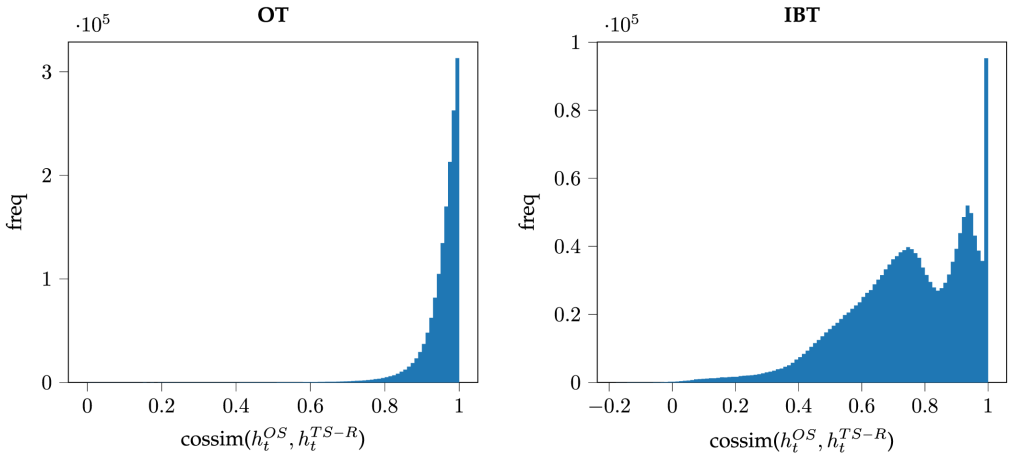
FIGURE 3.1: **Differences between embeddings for OT and IBT models.** We plot the histogram of cosine similarity between the **[CLS]** embedding produced by a model trained with the **OT** and **IBT** strategy under different intervention schemes. Results are reported with RoBERTa-large, and the figure is reproduced from Chaturvedi et al. [18].

ified **OS** dataset. Figure 3.1 plots the histogram of $cossim$ values for embeddings generated by RoBERTa-large. The sharply peaked distribution at 1 in Figure 3.1(a) indicates that there aren't major deviations from the **OS** dataset for the **TS** intervention following either training strategy; however, there is a stark difference between **OS** and **TS-R** (shown in Figure 3.1(b)) with the two training strategies. The drop in $cossim$ observed in Figure 3.1(b) indicates that the model modifies its embeddings in response to critical information deletion but maintains them despite large chunks of "irrelevant" information being removed, which is the desired behavior. This trend is not localized to the $[CLS]$ embedding, but a similar experiment working with every common token in **OS, TS**, and **TS-R** shows the same behavior (see Figure

3.2). This indicates that the **IBT** strategy successfully makes models more aligned to domain expectations while maintaining baseline performance.



(a) **OS** and **TS**



(b) **OS** and **TS-R**

FIGURE 3.2: **Differences between embeddings for OT and IBT models.** We plot the histogram of cosine similarity between the common token embeddings produced by a model trained with the **OT** and **IBT** strategy under different intervention schemes. Results are reported with RoBERTa-large, and the figure is reproduced from Chaturvedi et al. [18].

4

DOMAIN-AWARE LEARNING & EVALUATION

In this chapter[1], we tackle the problem of incorporating logic rules into the DL framework. As discussed in Chapter 2, this is a pressing challenge in several critical domains like healthcare, robotics, law, etc. Notably, these vital expert-dominated areas are also likely to present significant challenges from the data annotation perspective since expansive resources must be devoted to this endeavor. Since the deployment of successful DL models requires copious amounts of training data, the annotations in these settings are likely to contain significant quantities of label noise. This is the problem we tackle in Section 4.1. In particular, we explore how the presence of annotation noise affects domain knowledge adherence and propose a novel technique to exploit these noisy annotations to not only improve logical coherence but also improve learning performance.

This chapter (Section 4.2) also explores evaluating models and specifically looks at problems with common metrics used in DL with the goal of analyzing their efficacy from the perspective of domain knowledge obedience. Although standard metrics provide adequate signals in the vast majority of cases, each of them often has its own pitfalls [14]. Thus, to avoid arriving at a misleading perception of performance, the

[1]This chapter is largely based on our papers titled *"DOST–Domain Obedient Self-supervision for Trustworthy Multi Label Classification with Noisy Labels"* [149] and *"MedTric : A clinically applicable metric for evaluation of multi-label computational diagnostic systems"* [148].

accepted best practice is to report results on a multitude of metrics [56], which also presents challenges.

In Section 4.2.2, we set out a few basic ground rules a " *domain obedience*" metric must follow in a clinical setting, followed by an exploration of some common ML metrics and their potential issues. We then illustrate how to construct a metric from the ground up with the domain constraints in mind (Section 4.2.5). This is followed by a validation of the proposed metric in some key areas (Section 4.2.6). Although the designed metric targets a clinical use case, the process of arriving at the metric is generalizable to any target domain presenting similar demands.

4.1 DOMAIN KNOWLEDGE AUGMENTATION

Supervised DL algorithms have achieved significant success in addressing challenges across various domains. Despite this, they face a critical drawback: the vast amounts of annotated data required. This necessity often leads to annotation methods that can be unreliable and introduce **label noise**[2]. The issue becomes more problematic when applications demand specialized domain knowledge, as data labeling becomes an exceedingly resource-intensive task [41]. Consequently, datasets tend to be smaller [163] and may still contain substantial amounts of noise. Disagreements among experts over labels (inter-observer variability) and uncertainties that arise (annotator errors) are frequently overlooked. Annotations are often either automatically generated using algorithms or crowdsourced with the assistance of non-experts, resulting in quality degradation [99, 129]. Given that DL models are susceptible to overfitting and can even adapt to randomly assigned labels [214], label noise poses a significant challenge [99, 188] in creating dependable DL systems.

Despite MLC being a widespread problem across several domains [76, 211, 217, 220], this has not been widely studied in a noisy setting. As discussed in Chapter 2, this is also a significantly harder problem. Since an arbitrary number of classes must be predicted, the number of distinct annotations increases from $|A|$ to $\sim 2^{|A|}$ for target classes in A, making errors more likely. Moreover, in contrast to the single-label setting (MCC), where a mislabeled instance results in exactly two classes being flipped, the multi-label setting allows for more complex errors. For example, a subset or superset of the target classes may have been annotated, which still retains valuable

[2] In this context, we use the terms annotation and label interchangeably.

information. Therefore, it is not ideal to treat these instances in the same manner as completely erroneous entries.

In this chapter, we focus on annotation errors that **contradict domain rules**. In Chapter 2, we saw that DL systems do not natively abide by domain rules, and this phenomenon is exacerbated by noisy annotations, resulting in models that frequently generate incoherent predictions during inference. Such errors are more damaging than simple mislabeling, as they appear "*absurd*" from the perspective of domain experts, thus posing a significant barrier to widespread adoption.

A naive solution would be to simply eliminate these contradictory instances from the dataset following the available domain rules. However, since high-quality data is often limited, this should be the last resort. The question we aim to address in this chapter is: *can we do better?* Specifically, can we leverage domain rules to develop models that align more closely with those rules (avoiding inferences that contradict them) and perform as well as, or better than, models trained on a reduced dataset?

Our investigations start by assessing the impact of annotation noise on model performance, considering both standard metrics and compliance with domain rules. We then introduce a novel *domain-obedient self-supervised training* (DOST) paradigm that yields predictions that are more aligned with the rule set and surpasses the performance of models trained on the reduced dataset. This approach is more data-efficient and utilizes instances that violate domain rules to significantly mitigate the effects of label noise. Additionally, it imbues the model with semantic information from the domain, leading to enhanced performance.

4.1.1. RELATED WORKS

Label noise is a significant challenge in DL systems, prompting researchers to develop various solutions [88, 168]. Among these solutions are techniques such as label cleaning and pre-processing [186], network architecture modifications [182], and the implementation of robust loss functions and regularization strategies [53, 67]. While there is a substantial body of work addressing noise in the single-label (MCC) context, explorations in the multi-label context are less prevalent. The traditional approach in MLC involves decomposing the problem into a series of independent binary classification tasks. Interestingly, recent advancements in MLC have been reliant on leveraging label correlations [19, 194], which exacerbates the domain coherence issue in the presence of noise. To address this, Zhao and Gomez [218]

introduced a loss function for learning robust embeddings.

This issue is often approached through the lens of disambiguation, i.e., given a dataset with noisy annotations, the goal is to first recover the ground-truth labels from candidate labels before applying learning algorithms. For instance, Xie and Huang [202], and Fang and Zhang [45] explored the introduction of labeling confidences, whereas Sun et al. [174] applied a low-rank and sparse decomposition technique. Garcia et al. [52] suggested a meta-learning system that predicts the performance of noise filters in noisy data identification tasks. Furthermore, Xie and Huang [203] proposed a framework for MLC that concurrently identifies noisy labels.

Our research ventures into domain knowledge augmentation, particularly focusing on logical constraints derived from domain information. Logically constrained MLC is a well-explored area and has been shown to outperform conventional MLC systems in certain tasks. Giunchiglia and Lukasiewicz [57] developed a method for hierarchical MLC problems by adding a constraint layer to their models, which uses hierarchies in the annotations to improve performance. This method demonstrated superior results compared to standard post-processing techniques, which do not incorporate constraint information into the model. Melacci et al. [115] trained MLCs using an augmented loss function that encodes logic constraints as a polynomial of predicate probabilities, showing that this approach helps mitigate adversarial attacks.

To our knowledge, logically constrained MLC has not been investigated in a noisy label context, where such constraint violations are frequent. The method we propose, **DOST**, capitalizes on these logical constraints to detect noisy labels and concurrently enhance both performance and domain coherence.

4.1.2. DEFINITIONS AND NOTATION

We start with a dataset $\mathcal{D} = \{(x_i, y_i) | \, \forall i \in \{1, \ldots, N\}\}$ where $x_i \in \Lambda$ are the data instances and y_i are the corresponding (potentially erroneous) labels. We have a set of target classes $A = \{a_1, a_2, \ldots, a_p\}$, and we have $y_i \subseteq A \; \forall i \in \{1, \ldots, N\}$. Let a_j also denote a unary predicate, and $a_j(x_i)$ is `true`, if x_i is an instance of a_j else `false`. So, the ordered pair (x_i, y_i) induces the formula:

$$\begin{aligned} \mathrm{F}(x_i, y_i) = a_{\beta_1}(x_i) \wedge a_{\beta_2}(x_i) \wedge \ldots a_{\beta_t}(x_i) \wedge \\ \neg a_{\gamma_{t+1}}(x_i) \wedge \neg a_{\gamma_{t+2}}(x_i) \wedge \ldots \neg a_{\gamma_p}(x_i) \end{aligned} \tag{4.1}$$

where, $\forall j$, a_{β_j} are in y_i and a_{γ_j} are not in y_i. $\beta_1 < \ldots < \beta_t < \gamma_{t+1} < \ldots < \gamma_p$.

We also have a set of domain rules $\mathrm{R} = \{r_1, r_2, \ldots\}$, where each r_k takes the form $r_k = \forall x,\ \phi_k(x) \rightarrow \psi_k(x)$, where ϕ_k and ψ_k are first-order logic formulas built from the predicates in A and the logical connectives $\{\wedge, \vee, \neg\}$.

We call an annotation y_i of x_i **contradictory** or a **contradiction** (denoted $C(y_i)$) with the rules R if and only if:

$$\exists\, r \in \mathrm{R},\ \text{such that } r \Rightarrow \neg \mathrm{F}(x_i, y_i) \tag{4.2}$$

To measure the amount of contradiction in a set of annotations $Y = \{y_i | i \in \{1 \ldots n\}\}$, we use the metric $C(Y)$, which is defined by the following equation:

$$C(Y) = \frac{1}{|Y|} \sum_{i=1}^{n} \text{Number of rule violations in } y_i \tag{4.3}$$

With the domain rules R, we define two auxiliary datasets, $\mathcal{D}_C$ and $\hat{\mathcal{D}} = \mathcal{D} - \mathcal{D}_C$, where $\mathcal{D}_C$ is the subset of the dataset with contradictory annotations, i.e., $(x_i, y_i) \in \mathcal{D}_C \iff y_i$ is a contradictory annotation. Let $|\mathcal{D}_C| = N_C$, and $p_c = \frac{N_C}{N}$ represents the proportion of contradictory samples.

PROBLEM SETUP

Given dataset $\mathcal{D}$ and a rule set R, we want f_ν, such that:

$$\begin{aligned} f_\nu &: \Lambda \rightarrow [0,1]^p \\ [f_\nu(x)]_k &\approx P\big(a_k(x)\big) \end{aligned} \tag{4.4}$$

The MLC prediction y is then given by:

$$y = g_{\nu,\mu}(x) = \{a_k | [f_\nu(x)]_k \geq \mu,\ \forall a_k \in A\} \tag{4.5}$$

for some constant $\mu \in [0, 1]$. Ideally, we should have

$$\begin{aligned} P\Big(a_k \in g_{\nu,\mu}(x) \Big| a_k(x)\Big) &\rightarrow 1 \\ P\Big(a_k \in g_{\nu,\mu}(x) \Big| \neg a_k(x)\Big) &\rightarrow 0 \\ P\Big(C\big(g_{\nu,\mu}(x)\big)\Big) &\rightarrow 0 \end{aligned}$$

Or, simply put, we want to have an accurate multi-label classifier that avoids contradictory annotations.

In the ensuing discussion, we only consider rules of the form $\forall x, a_i(x) \Rightarrow \neg\big(a_{\beta_1}(x) \vee a_{\beta_2}(x) \vee \ldots a_{\beta_t}(x)\big)$ for $a_{\beta_l} \in A$ $(\beta_l \neq i)$ in order to simplify creation of R.

4.1.3. DOST ALGORITHM

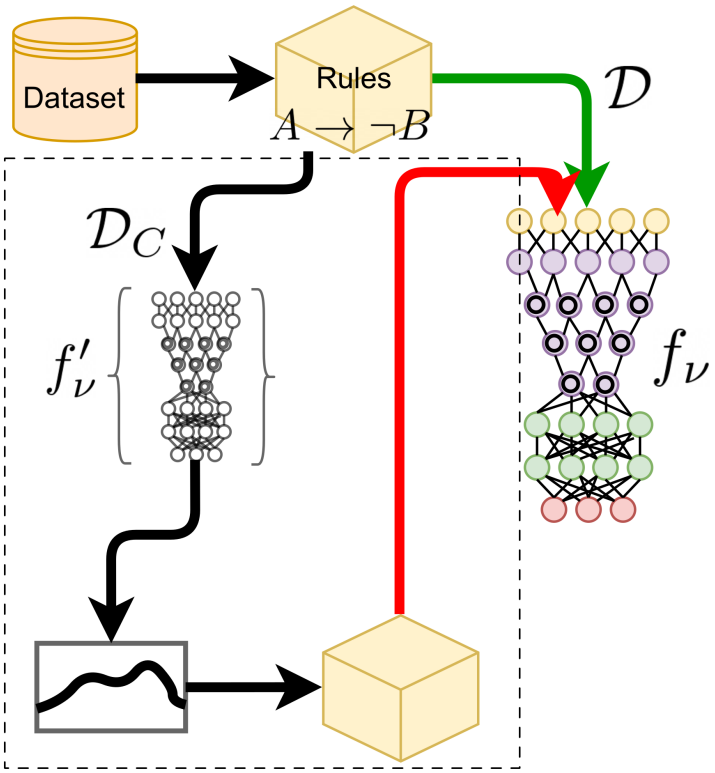


FIGURE 4.1: **Schematic diagram of DOST algorithm**
The dataset is partitioned based on rules in R, and the resulting noisy samples are passed on to an older copy of the model. The predictions from the older copy (f'_ν) are then used to compute potential conflicts, which are then used as negative samples. Figure is reproduced from Saha et al. [149].

In this section we outline **DOST**, the **D**omain **O**bedient **S**elf-supervised **T**raining algorithm, in order to find $g_{\nu,\mu}$ such that $P(C(g_{\nu,\mu}))$ is minimized alongside accurate identification of the classification targets. Given $\mathcal{D}$ and R we construct $\hat{\mathcal{D}}, \mathcal{D}_C$ and we define a rule matrix $\rho \in \mathbb{R}^{p \times p}$, as follows:

$$[\rho]_{ij} = \begin{cases} 1 & \text{if } \exists\ r \in \mathrm{R},\ \text{s.t. } r \Rightarrow\ \forall\ x \in \Lambda,\ a_i(x) \Rightarrow \neg a_j(x) \\ 0 & \text{otherwise.} \end{cases} \tag{4.6}$$

With $\hat{\mathcal{D}}$, $\mathcal{D}_C$, and ρ, we train a deep neural network (with sigmoid outputs) on $\hat{\mathcal{D}}$ as usual (with BCE loss); and for samples in $\mathcal{D}_C$ we use the target

$$\delta_C(x_i) = \rho\Big[\arg\max\big(f'_\mu(x_i)\big)\Big] \tag{4.7}$$

$$\mathcal{L}_C(x_i) = -\zeta \cdot \sum_{j=1}^{p}\Big[[\delta_C(x_i)]_j \cdot \log\big(1 - [f_\mu(x_i)]_j\big)\Big] \tag{4.8}$$

where ζ is a constant (hyper-parameter), and f_μ represents the deep network with parameters μ. f'_μ is a slightly older copy of the network (e.g., from the preceding epoch) with parameters (μ') treated as constants. Thus, in effect, we find potential classes that result in rule violations based on the old model's best guess, and we use those as negative examples while training. Since $\delta_C(x_i)$ is not a function of μ, $\mathcal{L}_C$ is differentiable and can be used with any standard gradient based training method. The algorithm is outlined in Algorithm 1, and a schematic diagram is given in Figure 4.1.

Algorithm 1 : DOST

while not StopCondition **do**
  Get a batch of samples from $\mathcal{D}$
  $l \leftarrow 0$
  for $(x_i, \hat{y}_i)$ in batch **do**
    if $(x_i, y_i) \in \hat{\mathcal{D}}$ **then**
      $l \mathrel{+}= \mathcal{L}\Big(\hat{y}_i, f_\mu(x_i)\Big)$ ▷ Classification loss
    else
      $l \mathrel{+}= \mathcal{L}_C(x_i)$ ▷ Eq. 4.8
    end if
  end for
  Back-propagate(l)
  Update model μ
  Every k steps $f' \leftarrow f$
  if ... **then** ▷ Check if model converged
    StopCondition $\leftarrow$ True
  end if
end while

Although we discuss rules of the form $\forall x,\ a_i(x) \Rightarrow \neg\big(a_{\beta_1}(x) \vee a_{\beta_2}(x) \vee \ldots a_{\beta_t}(x)\big)$, the **DOST** framework is more general. For example, consider a rule of the form $\forall x,\ a_{i_1}(x) \wedge a_{i_2}(x) \wedge \ldots \wedge a_{i_k}(x) \Rightarrow a_\beta(x)$. Here we can modify Equation 4.7 to be $\mathrm{top}_k(f'_\mu(x_i))$ and construct an appropriate tensor ρ. All such terms can then be added to Equation 4.8 (with appropriate signs) to incorporate rules of this nature. $\mathcal{L}_C$

is not necessarily limited to application on $\mathcal{D}_C$, and with an appropriately chosen ζ, can serve as an auxiliary loss to the standard BCE loss in order to mitigate rule violations.

4.1.4. EXPERIMENTS

EXPERIMENTAL DETAILS

We conducted experiments using two extensive multi-label datasets: PASCAL VOC [43] and PhysioNet 2020/21 [3]. The PhysioNet 2020/21 dataset consists of 12-lead ECG signals pooled from several publicly available datasets. Each one of them is marked with a subset of diagnoses from a set of 27 possible diagnoses. A rule set was constructed with inputs from experts in the field and is given in Table 4.1.

TABLE 4.1: **Contradictory pairs in PhysioNet dataset [3].**
Each pair (A, B) represents a pair of rules of the form $\forall x, A(x) \Rightarrow \neg B(x)$ and $B(x) \Rightarrow \neg A(x)$. The set of all these rules is R.

(AF , IAVB)	(PR , AF)	(PR , AFL)	(PR , CRBBB)
(PR , IRBBB)	(PR , LAnFB)	(PAC , AF)	(PAC , AFL)
(LPR , AF)	(LPR , AFL)	(SA , AF)	(SA , AFL)
(SA , PR)	(SB , AF)	(SB , AFL)	(SB , PR)
(NSR , IAVB)	(NSR , AF)	(NSR , AFL)	(NSR , Brady)
(NSR , CRBBB)	(NSR , IRBBB)	(NSR , LAnFB)	(NSR , LAD)
(NSR , NSIVCB)	(NSR , PAC)	(NSR , PVC)	(NSR , LPR)
(NSR , LQT)	(NSR , QAb)	(NSR , RAD)	(NSR , SA)
(STach , AF)	(STach , AFL)	(STach , Brady)	(STach , PR)
(STach , SB)	(STach , NSR)	(TAb , NSR)	(TInv , NSR)
(NSR , LBBB)	(NSR , SB)		

PASCAL VOC serves as a widely recognized benchmark in computer vision for MLC, where the task is to identify a subset of classes present in an image from a possible 20 classes. Although PASCAL VOC does not inherently include rules prohibiting specific class combinations, we devised a rule set for demonstration purposes based on classes that do not co-occur in the training data (see Table 4.2).

By constructing the rule set based on the class co-occurrences within the datasets, we obviously find that PASCAL VOC annotations do not contradict the rule set. In contrast, about 20% of the annotations in the PhysioNet dataset conflicted with the established rule set. To assess the impact of noise on model performance, we

TABLE 4.2: Rule set R used with the PASCAL-VOC dataset [43]

$\forall$ image $x \in \mathcal{D}$, we have	
$\texttt{chair}(x) \rightarrow$	$\neg(\texttt{bus}(x) \vee \texttt{dog}(x) \vee \texttt{train}(x) \vee \texttt{pottedplant}(x) \vee \texttt{sofa}(x) \vee \texttt{bird}(x) \vee \texttt{cat}(x) \vee \texttt{tvmonitor}(x) \vee \texttt{motorbike}(x) \vee \texttt{person}(x) \vee \texttt{sheep}(x) \vee \texttt{bottle}(x) \vee \texttt{bicycle}(x) \vee \texttt{aeroplane}(x) \vee \texttt{horse}(x) \vee \texttt{diningtable}(x) \vee \texttt{car}(x) \vee \texttt{boat}(x) \vee \texttt{cow}(x))$
$\texttt{bus}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{sofa}(x) \vee \texttt{bird}(x) \vee \texttt{cat}(x) \vee \texttt{tvmonitor}(x) \vee \texttt{horse}(x) \vee \texttt{diningtable}(x) \vee \texttt{cow}(x))$
$\texttt{dog}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{train}(x) \vee \texttt{aeroplane}(x))$
$\texttt{train}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{dog}(x) \vee \texttt{bird}(x) \vee \texttt{cat}(x) \vee \texttt{tvmonitor}(x) \vee \texttt{motorbike}(x) \vee \texttt{sheep}(x) \vee \texttt{bottle}(x) \vee \texttt{aeroplane}(x) \vee \texttt{diningtable}(x) \vee \texttt{cow}(x))$
$\texttt{pottedplant}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{sheep}(x) \vee \texttt{cow}(x))$
$\texttt{sofa}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{bus}(x) \vee \texttt{bird}(x) \vee \texttt{sheep}(x) \vee \texttt{aeroplane}(x) \vee \texttt{horse}(x) \vee \texttt{cow}(x))$
$\texttt{bird}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{bus}(x) \vee \texttt{train}(x) \vee \texttt{sofa}(x) \vee \texttt{motorbike}(x))$
$\texttt{cat}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{bus}(x) \vee \texttt{train}(x) \vee \texttt{motorbike}(x) \vee \texttt{aeroplane}(x) \vee \texttt{horse}(x) \vee \texttt{boat}(x))$
$\texttt{tvmonitor}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{bus}(x) \vee \texttt{train}(x) \vee \texttt{motorbike}(x) \vee \texttt{sheep}(x) \vee \texttt{aeroplane}(x) \vee \texttt{cow}(x))$
$\texttt{motorbike}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{train}(x) \vee \texttt{bird}(x) \vee \texttt{cat}(x) \vee \texttt{tvmonitor}(x) \vee \texttt{diningtable}(x))$
$\texttt{person}(x) \rightarrow$	$\neg(\texttt{chair}(x))$
$\texttt{sheep}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{train}(x) \vee \texttt{pottedplant}(x) \vee \texttt{sofa}(x) \vee \texttt{tvmonitor}(x) \vee \texttt{bicycle}(x) \vee \texttt{aeroplane}(x) \vee \texttt{diningtable}(x) \vee \texttt{boat}(x))$
$\texttt{bottle}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{train}(x) \vee \texttt{aeroplane}(x) \vee \texttt{cow}(x))$
$\texttt{bicycle}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{sheep}(x) \vee \texttt{aeroplane}(x) \vee \texttt{horse}(x))$
$\texttt{aeroplane}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{dog}(x) \vee \texttt{train}(x) \vee \texttt{sofa}(x) \vee \texttt{cat}(x) \vee \texttt{tvmonitor}(x) \vee \texttt{sheep}(x) \vee \texttt{bottle}(x) \vee \texttt{bicycle}(x) \vee \texttt{horse}(x) \vee \texttt{diningtable}(x) \vee \texttt{cow}(x))$
$\texttt{horse}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{bus}(x) \vee \texttt{sofa}(x) \vee \texttt{cat}(x) \vee \texttt{bicycle}(x) \vee \texttt{aeroplane}(x) \vee \texttt{diningtable}(x) \vee \texttt{boat}(x))$
$\texttt{diningtable}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{bus}(x) \vee \texttt{train}(x) \vee \texttt{motorbike}(x) \vee \texttt{sheep}(x) \vee \texttt{aeroplane}(x) \vee \texttt{horse}(x) \vee \texttt{cow}(x))$
$\texttt{car}(x) \rightarrow$	$\neg(\texttt{chair}(x))$
$\texttt{boat}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{cat}(x) \vee \texttt{sheep}(x) \vee \texttt{horse}(x))$
$\texttt{cow}(x) \rightarrow$	$\neg(\texttt{chair}(x) \vee \texttt{bus}(x) \vee \texttt{train}(x) \vee \texttt{pottedplant}(x) \vee \texttt{sofa}(x) \vee \texttt{tvmonitor}(x) \vee \texttt{bottle}(x) \vee \texttt{aeroplane}(x) \vee \texttt{diningtable}(x))$

introduced additional noise in both datasets.

Noisy instances were introduced with a probability p_C, either by selecting from a pool of pre-defined noisy instances or by injecting noise according to a geometric distribution. Consequently, from the dataset $\mathcal{D}$ and the rule set R, we generated the datasets $\hat{\mathcal{D}}$ and $\mathcal{D}_C$ such that $p_C = \frac{|\mathcal{D}_C|}{|\mathcal{D}|}$. These datasets were then used to evaluate the influence of noise and the efficacy of various noise mitigation strategies.

The experiments were executed on a single NVIDIA RTX A6000 48GB GPU, and results are reported using 10-fold cross-validation across all scenarios.

RESULTS

Just like we noted in Section 2.3, we observe that, even without any noisy annotations in the dataset, standard DL algorithms still produce contradictory predictions (see Table 4.3). Our fine-tuned ResNet50 pre-trained [65] on ImageNet [28] for the PASCAL VOC task was comparatively better than the model trained on PhysioNet, perhaps owing to the large-scale pre-training.

Our experimental results consistently show that increasing the percentage of noisy annotations negatively impacts both the quantity of contradictions and overall model

TABLE 4.3: **DL model trained on clean dataset produces contradictions.**
We report 10-fold cross-validation $C(Y)$ results (see Equation 4.3).

	PhysioNet 2020	**PASCAL VOC**
Train	4.02% ±1.65	0.05% ±0.04
Test	3.57% ±1.32	0.11% ±0.09

performance (see Figure 4.2).

Subsequently, we evaluated the impact of two mitigation strategies on contradictory outputs: the naive approach of filtering out conflicting annotations and our proposed **DOST** paradigm (see Figure 4.3). Since we apply the self-supervision loss update solely on $\mathcal{D}_C$ (see Equation 4.8, Algorithm 1), the two strategies should yield similar outcomes when $\mathcal{D}_C$ is small. We observe that DOST significantly reduces contradictory predictions, particularly at higher noise levels.

As previously mentioned, the fine-tuned ResNet50 model applied to the PASCAL VOC task exhibits fewer contradictory predictions, and this remains stable throughout the filtering strategy. Although slight improvements with DOST are noted at higher noise levels, the difference is marginal. The effect is more substantial in the model trained from scratch on the PhysioNet dataset, where DOST effectively reduces contradictory outputs to levels better than baseline in most cases (see Table 4.4).

TABLE 4.4: **Performance of DOST at various noise levels.**
DOST eliminates contradictory outputs at high noise levels and even outperforms the ideal no-noise scenario. We report $C(Y)$ per hundred instances from the PhysioNet dataset. Perfect dataset refers to a dataset with 0% noise level.

Noise level	Perfect dataset	**DOST**	Filtered dataset	Noisy dataset
12.5%	3.57 ±1.32	3.82 ±0.41	4.42 ±1.67	14.90 ±4.38
25%		**2.46** ±1.19	4.22 ±1.78	18.09 ±3.31
37.5%		**1.36** ±0.61	5.67 ±2.33	21.61 ±2.86
50%		**1.61** ±0.56	10.07 ±4.10	23.66 ±3.98

Lastly, we assessed the influence of both strategies on overall performance. The **DOST** paradigm surpasses the naive filtering strategy, showing notable improvements across various metrics (see Figure 4.4). In particular, with the PASCAL VOC dataset, the DOST paradigm almost entirely mitigates the effects of noise (see Table 4.5).

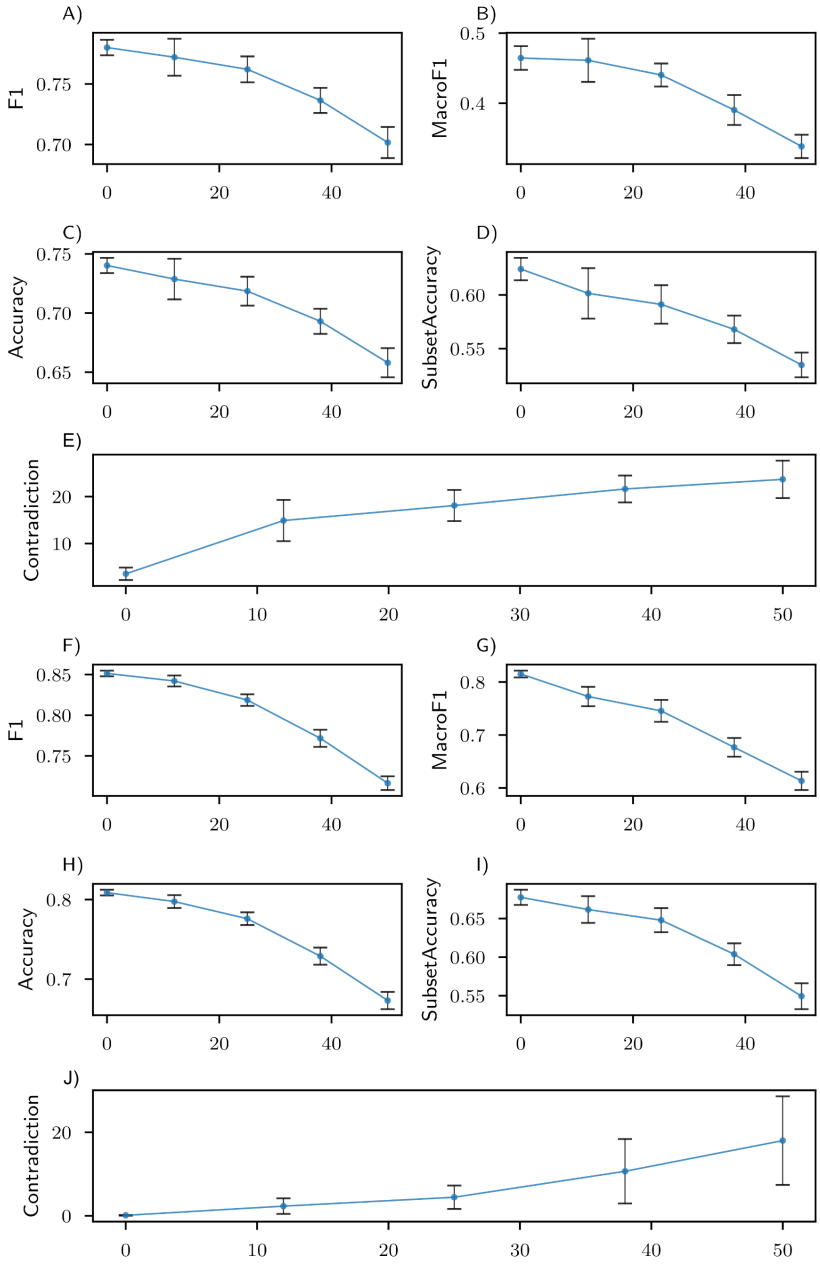


FIGURE 4.2: **Effect of noise on performance of DL models.**
With increasing amounts of noise model performance continually degrades, whereas contradictions increase. The X-axis represents the percentage of training instances containing contradictions. (A-E) shows various metrics for PhysioNet dataset, and (F-J) for PASCAL VOC.

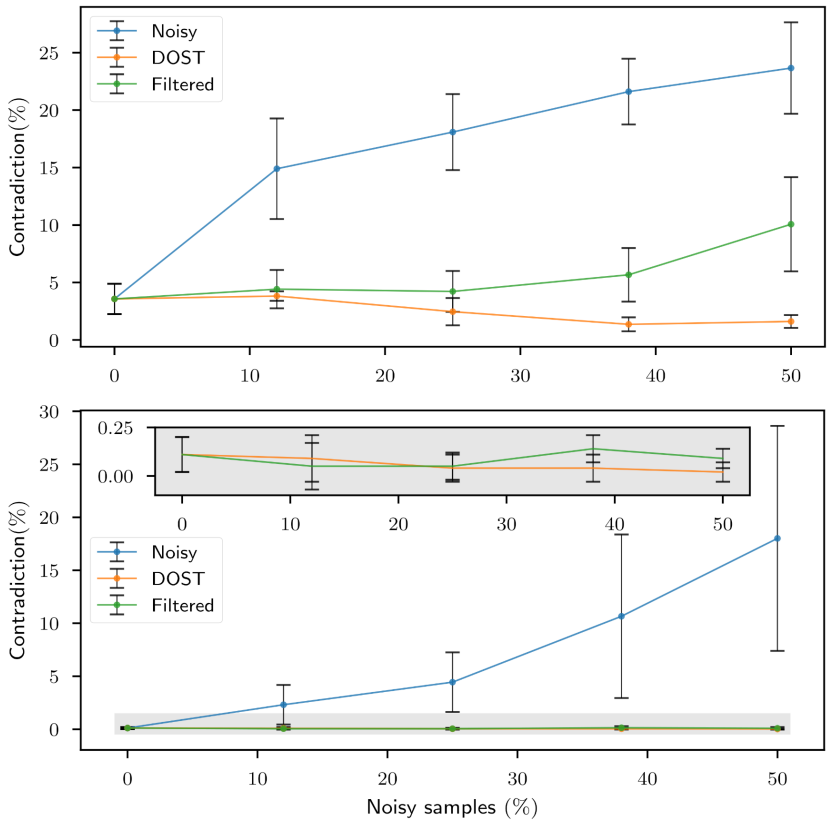


FIGURE 4.3: **DOST paradigm significantly reduces rule violations**
The *top* panel presents results from the PhysioNet dataset and the *bottom* panel presents results from PASCAL-VOC. The shaded region in the figure below is a magnified view of the curves close to the X-axis. Figure is reproduced from Saha et al. [149].

TABLE 4.5: Even with 50% of instances containing noisy labels, DOST successfully counteracts the effect of noise. Filtered dataset refers to the dataset with erroneously annotated samples removed, i.e., it is smaller in size.

Metric	Perfect Dataset	**DOST**	Noisy Dataset	Filtered Dataset
Noise Level	0%	50%	50%	0%
Accuracy	0.809 ±0.004	0.791 ±0.008	0.673 ±0.011	0.771 ±0.008
F1	0.851 ±0.004	0.836 ±0.008	0.716 ±0.008	0.820 ±0.007
Macro F1	0.815 ±0.007	0.798 ±0.011	0.613 ±0.017	0.768 ±0.007
Subset Accuracy	0.677 ±0.010	0.656 ±0.010	0.549 ±0.017	0.621 ±0.017

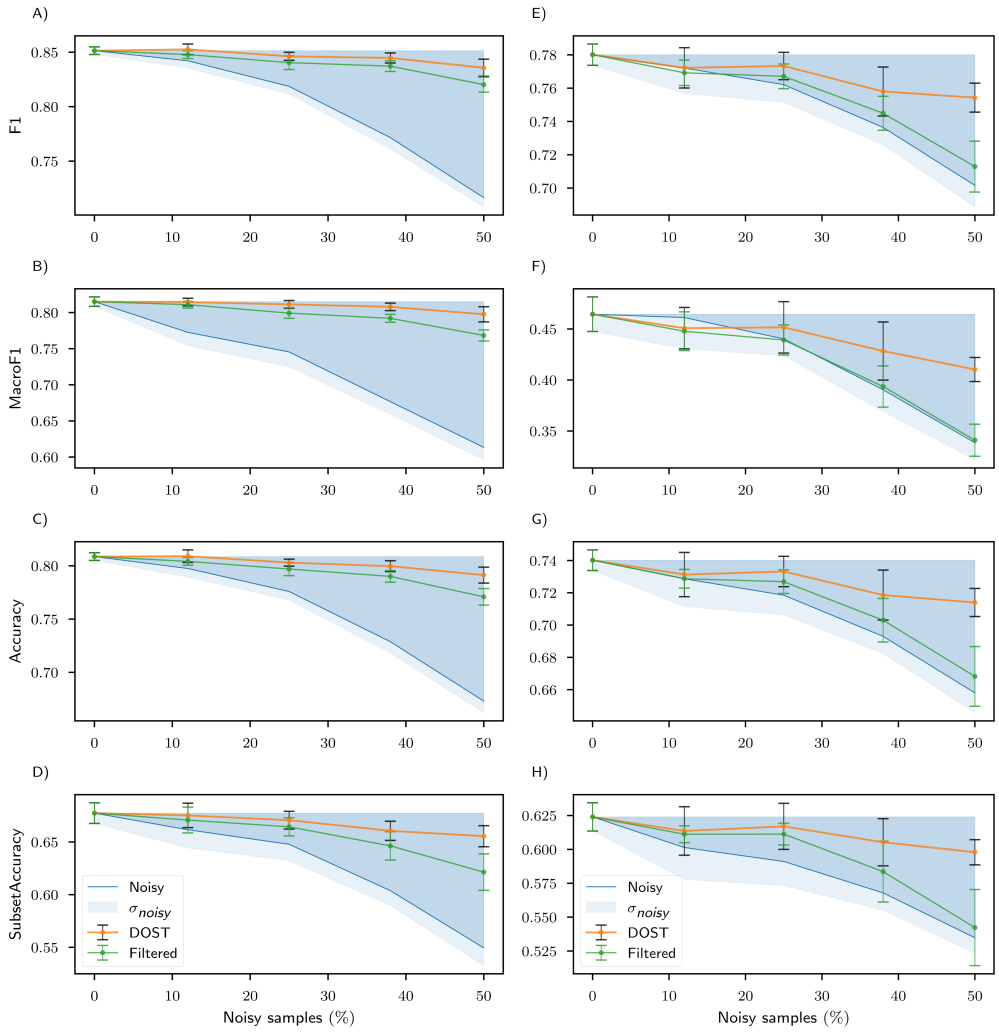


FIGURE 4.4: **DOST paradigm improves performance across several metrics.** The shaded region above the blue line (performance of the model trained on a noisy dataset) is the potential room for improvement, given a hypothetical clean dataset, which is usually not available in practice. **DOST** outperforms the naive filtering approach in all cases, and on the PASCAL VOC dataset, almost manages to counteract the effect of noise altogether. The *left* (A-D) panel consists of plots of various metrics for PASCAL VOC, and the *right* (E-H) panel plots the same for PhysioNet. Figure is reproduced from Saha et al. [149].

4.2 EVALUATING MODELS

In this section, we look at some commonly used MLC metrics and the potential issues they present.

4.2.1. DEFINITIONS

We first present some definitions that are used throughout the rest of the chapter. Some of these definitions are also summarized in Table 4.6.

TABLE 4.6: Common notation associated with MLC.

Symbol	Meaning
A	Set of all possible classes (finite) $\lvert\mathrm{A}\rvert = P$
a_j	j-th element of A. $a_j \in \mathrm{A}\ \forall j \in \{1, \dots P\}$
2^{A}	The set of all possible subsets of A.
$\mathcal{D}$	The dataset (see Definition 4.1). $\lvert\mathcal{D}\rvert = N$
$(x_i, y_i) \in \mathcal{D}$	The i-th instance from the dataset with data $x_i \in \mathcal{R}$ and label $y_i \in 2^{\mathrm{A}}$.
$f_\theta(.)$	The prediction system parametrized by θ.
$\hat{z}_i \in 2^{\mathrm{A}}$	The prediction set for (x_i, y_i) given by f_θ
$\mathcal{M}(.,.)$	A metric; maps $\{(\hat{z}_i, y_i)\}$ to a score $s \in \mathbb{R}$

DEFINITION 4.1 (Dataset) *Given a set of samples and their respective annotations* $\mathcal{D} = \{(x_i, y_i)|\ \ i \in 1, \cdots, N\}$*, where* x_i *and* y_i *are the* i^{th} *sample and label, respectively is called the* ***dataset****. Each* y_i *is a set of classes (drawn from a fixed set of possible classes* $\mathrm{A} = \{a_1, a_2, \dots a_P\}$*), i.e.,* $y_i \in 2^{\mathrm{A}}$*.* $x_i \in \mathbb{R}^\chi$*.*

DEFINITION 4.2 (Classifier) $f_\theta : \mathbb{R}^\chi \to 2^{\mathrm{A}}$ *is called a classifier. Given a sample* x_i*, it attempts to recreate the corresponding label* $y_i \approx \hat{z}_i = f_\theta(x_i)$ *for some (potentially hidden) parameters* θ*.*

The results from the classification framework are often presented as scores, which relate to the likelihood of a particular class being identified, i.e., $g_\theta : \mathbb{R}^\chi \to [0, 1]^P$ and some thresholding protocol must be additionally specified. Generally, such a threshold $t_{\phi,\mathcal{D}}(x_i) \mapsto t_i \in [0, 1]^P$. A successful scheme should have:

$$[g_\theta(x_i)]_j > [g_\theta(x_i)]_k \text{ if } a_j \in y_i \text{ and } a_k \notin y_i \tag{4.9}$$

We also define

$$[z_i]_k = [h_\theta(x_i)]_k := \begin{cases} 1 \text{ if } [g_\theta(x_i)]_k > [t_i]_k \\ 0, \text{ otherwise} \end{cases} \tag{4.10}$$

The prediction set $\hat{z}_i$, i.e., the set of all label classes meeting the prediction threshold, is given by $\hat{z}_i = \{a_j | z_{ij} = 1,\ \forall j \in \{1, \ldots, P\}\}$. The combination of the function g_θ and the thresholding procedure results in what we refer to as a *classifier*.

In this chapter, we focus solely on evaluating the output set $\hat{z}_i$, in order to have the most general treatment of various kinds of ML-based classification systems. The aforementioned classification system, which includes the function g_θ and the thresholding strategy, is treated as a black box.

Note, this excludes metrics like AUC [131], etc., which, although useful in certain contexts, pose the additional challenge of requiring access to the implementation details of the classifier in question. For instance, a clever inference algorithm might make the decision to detect a class based on some accessory information in addition to the classifier outputs in a manner that abides by domain rules. Thus, to analyze the efficacy of an end-to-end system with regard to domain constraint adherence, it makes more sense to talk about the complete system. Thus, we restrict ourselves to implementation blind metrics, i.e., those that can be computed given just the output and target labels.

DEFINITION 4.3 (Wrong Classification) *A prediction $\hat{z}_i$ is said to be a wrong classification if $\hat{z}_i \cap y_i = \phi$, i.e., prediction and ground truth are disjoint.*

DEFINITION 4.4 (Missed Classification) *A prediction $\hat{z}_i$ is said to be a missed classification if $\hat{z}_i \subsetneq y_i$, i.e., prediction is a proper subset of ground truth labels.*

DEFINITION 4.5 (Over-Classification) *A prediction $\hat{z}_i$ is said to be an over-classification, if $y_i \subsetneq \hat{z}_i$, i.e., ground truth is a proper subset of predicted labels.*

DEFINITION 4.6 *The elements of sets $\hat{z}_i - y_i, y_i - \hat{z}_i,$ and $y_i \cap \hat{z}_i$ are called extra predictions, missed predictions, and correct predictions respectively.*

4.2.2. DIAGNOSTIC CONSTRAINTS

ML techniques have shown great promise in computational diagnostics [62, 221] and have been applied to a wide set of diagnostic problems [63, 219], which are

often *multi-label*, i.e., where several diagnostic features might be detected from one data sample [192]. For example, consider a blood sample that might be evaluated by a pathologist to detect the presence of several pathogens or a radiologist marking various anomalies in a CT scan.

From an aggregated health-care cost perspective, the potential benefit from algorithmic screening can be massive (see Figure 4.5), provided we can find a suitable system. Therefore, comparing several competing computational diagnostic systems in accordance with clinical outcomes is paramount for deployment in clinical applications. This however continues to pose a challenge.

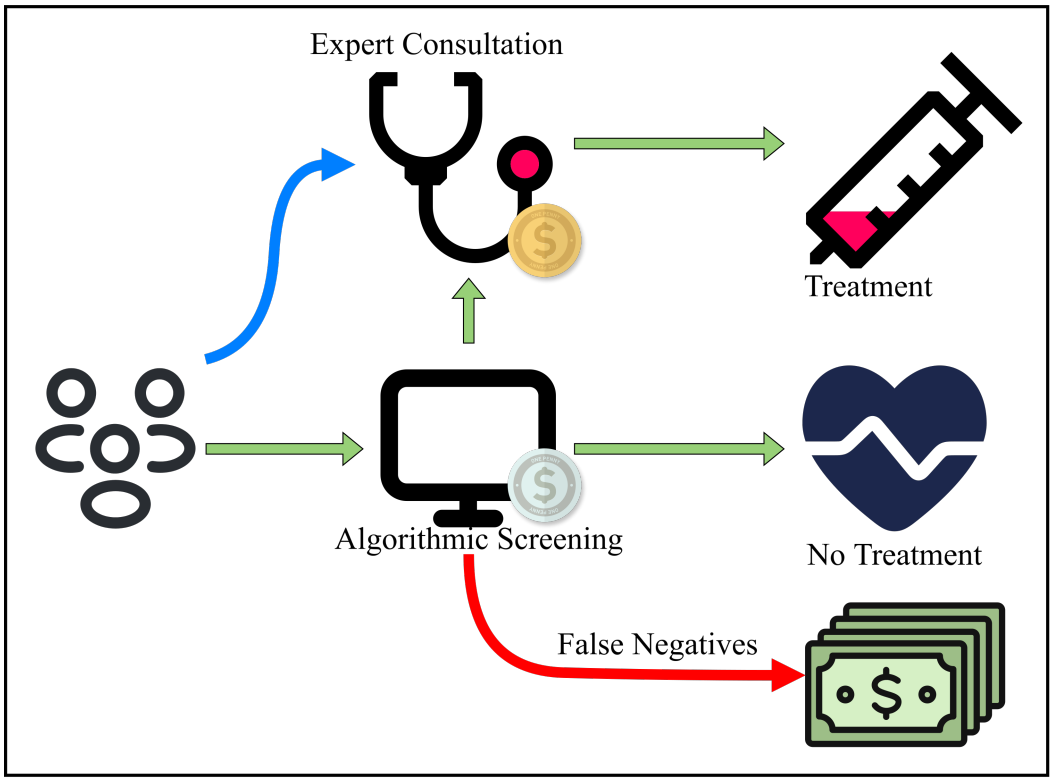


FIGURE 4.5: **Algorithmic screening cuts down on health-care costs.**
Since algorithmic screening is orders of magnitude cheaper than expert intervention, well-designed computational systems can make health care accessible to a larger population.

There is currently no agreement on the optimal metric for evaluating models [219], and it is generally advised to asses models using multiple metrics [56]. Advances in multi-label diagnostics often have to contend with this challenge [17, 63, 219]. This scattershot approach, however, introduces complications, as comparing models evaluated with different sets of metrics becomes problematic [131]. Furthermore, selecting a particular metric can emphasize a model's strengths while concealing

its weaknesses [131]. Different metrics often provide conflicting evaluations of system performance [87], meaning the choice of an optimal diagnostic system can be dictated by the choice of the metric. Even when results are presented across multiple metrics, they may not sufficiently inform clinical decision-making. A broad array of scores, each reflecting various performance aspects, may not address the question, "*Which system is more suitable for clinical use?*" Since these metrics are adapted from machine learning, where priorities differ, a higher metric score does not necessarily signify superior diagnostic performance, and vice versa. Therefore, it is crucial to develop a metric that can effectively rank computational diagnostic models based on relevant clinical outcomes [3].

In clinical practice, some facts are ubiquitous and can be treated like axioms. For instance, a wrong diagnosis (wrong classification) is worse than a missed diagnosis (missed classification), which is in turn worse than over-diagnosis (over-classification) up to a certain extent. The standard metrics used in an MLC setting do not reflect this.

Also, certain sets of diagnostics have similar treatment plans and outcomes [3], thus making certain types of missed diagnosis less deleterious. Additionally, if there are k possible diagnoses, all 2^k might not be feasible or logically sound. For example, consider that a patient cannot simultaneously demonstrate sinus tachycardia (elevated heart rate) and sinus bradycardia (lowered heart rate) or hypo- and hypertension. In general, given k classes, there may be a number of first-order logic rules that preclude or imply the presence of other classes under certain conditions–and a successful diagnostic system must adhere to them strictly.

In a computational diagnostic system, the principle of risk avoidance suggests that sensitivity should be linked to cost, meaning that more serious conditions should be detected with higher sensitivity than less significant issues. However, increasing sensitivity often reduces specificity, potentially leading to alarm fatigue. Consequently, a general MLC metric might not reflect the clinical principles and practices critical for evaluating such a system. It may fail to capture essential characteristics necessary for a diagnostic tool.

With clinical considerations in mind and in consultation with domain experts, we have identified the key attributes that a clinically aligned metric should exhibit.

- Missed diagnosis is more harmful than over-diagnosis.
- A wrong diagnosis is more harmful than an over-diagnosis or missed diagnosis.

- Some diagnoses have more clinical significance.
- Some diagnoses are contradictory and should be disqualifying.
- The quality of a diagnostic tool should not depend on the relative proportions of diseases present in the population (dataset distribution independence).

In the following sections, we will use these axioms and the rules posed in a diagnostic context to analyze various commonly used metrics and attempt to develop our own metric adhering to these principles.

4.2.3. METRICS

To judge the quality of the classifier f_θ over the dataset $\mathcal{D}$, it is sufficient to analyze the set $\mathcal{P} = \{(\hat{z}_i, y_i) | \forall i, \text{ s.t. } (x_i, y_i) \in \mathcal{D}\}$. The job of a metric, given such a set $\mathcal{P}$, is to provide a number, which is correlated to the performance of the classification system. We shall not be exploring metrics designed for the label ranking task [111] (coverage, etc.); since they are not relevant in this context, instead we shall focus on *bipartition*-based metrics in the ensuing discussion, which are designed for the task at hand.

Bipartition metrics can be broadly divided into two categories: *label-based* (see Table 4.7a) and *example-based* (see Table 4.7b). The *example-based* metrics assign a score based on averages over certain functions of the actual and predicted label sets. *Label-based* metrics on the other hand compute the prediction performance of each label in isolation and then compute averages over labels.

Certain other binary metrics have been proposed in an MLC context, like the Mathews Correlation Coefficient [21], etc., and we can define macro/micro averages or example-based metrics based on these; however due to their limited usage in an MLC context, we omit these. Their definitions suggest that their behavior in key aspects follows the other metrics [56] discussed in the following section.

LABEL-BASED METRICS

Label-based metrics [111] typically use micro- or macro-averages of binary classification metrics like precision, recall, and F_1 (or the more general F_β) to summarize performance across multiple categories (see Table 4.7a). Specificity alone is not ideal for settings with class imbalance, a common challenge in many diagnostic datasets [151].

A *macro-averaged* metric is calculated by independently evaluating the binary metric for each class and then averaging over all classes. Conversely, a *micro-*

TABLE 4.7: Common example and label-based metrics in MLC.

Metric	Definition
Macro-precision	$\frac{1}{P}\sum_{j=1}^{P}\frac{tp_j}{tp_j+fp_j}$
Macro-recall	$\frac{1}{P}\sum_{j=1}^{P}\frac{tp_j}{tp_j+fn_j}$
Macro-F1-score	$\frac{1}{P}\sum_{j=1}^{P}F1_j, F1_j=\frac{2p_jr_j}{p_j+r_j}$
Micro-precision	$\frac{\sum_{j=1}^{P}tp_j}{\sum_{j=1}^{P}tp_j+\sum_{j=1}^{P}fp_j}$
Micro-recall	$\frac{\sum_{j=1}^{P}tp_j}{\sum_{j=1}^{P}tp_j+\sum_{j=1}^{P}fn_j}$
Micro-F1-score	$\frac{2\cdot\text{micro-precision}\cdot\text{micro-recall}}{\text{micro-precision}+\text{micro-recall}}$

(a) Label-based metrics.

Metric	Definition
Hamming Loss	$\frac{1}{N}\sum_{i=1}^{N}\frac{1}{P}\lvert\hat{z}_i\Delta y_i\rvert$
Accuracy	$\frac{1}{N}\sum_{i=1}^{N}\frac{\lvert\hat{z}_i\cap y_i\rvert}{\lvert\hat{z}_i\cup y_i\rvert}$
Precision	$\frac{1}{N}\sum_{i=1}^{N}\frac{\lvert\hat{z}_i\cap y_i\rvert}{\lvert y_i\rvert}$
Recall	$\frac{1}{N}\sum_{i=1}^{N}\frac{\lvert\hat{z}_i\cap y_i\rvert}{\lvert\hat{z}_i\rvert}$
F1-score	$\frac{1}{N}\sum_{i=1}^{N}\frac{2\times\lvert\hat{z}_i\cap y_i\rvert}{\lvert\hat{z}_i\rvert+\lvert y_i\rvert}$
Subset Accuracy	$\frac{1}{N}\sum_{i=1}^{N}I(\hat{z}_i=y_i)$

(b) Example-based metrics.

averaged metric aggregates the statistics across all classes before computing the final metric. Both methods, however, have inherent limitations. Micro-averaging tends to favor classifiers that perform well on the abundant classes, while macro-averaging benefits classifiers that excel in detecting rare classes. In clinical contexts, where certain presentations are infrequent but critical, micro-averaged measures are less meaningful, as rare conditions are often a cause for concern and might benefit greatly from intervention. Problems arise when the majority class is the primary cause for concern since a macro-averaged metric might give an overly positive impression of the diagnostic system's performance.

EXAMPLE-BASED METRICS

Example-based metrics [153] (Table 4.7b), are specifically crafted to highlight certain critical aspects of a multi-label classifier. It is generally insufficient to rely on just one or two metrics [219], as each offers unique insights that can be valuable.

A noteworthy recent effort by Alday et al. [3] aimed to develop a metric that incorporates clinical outcomes in a multi-label diagnostic context. This metric was designed to assess various computational models, which are tasked with identifying a subset of diagnostic features from 12-lead ECG signals across 27 potential diagnostic classes, many of which may coexist simultaneously. In this metric, we first define

the multi-class confusion matrix a as:

$$[a]_{jk} = \sum_{i=1}^{N} [a]_{\mathbf{i}jk}, \text{ where,} \tag{4.11}$$

$$[a]_{\mathbf{i}jk} = \begin{cases} \frac{1}{|\{\hat{z}_i \cup y_i\}|}, & \text{if } c_k \in \hat{z}_i \text{ and } c_j \in y_i \\ 0, \text{ otherwise.} \end{cases} \tag{4.12}$$

Next we compute $t(Y, Z) = \sum_k \sum_j [w]_{jk}[a]_{jk}$ where $[w]_{jk}$ is the weight matrix giving partial rewards to incorrect guesses. $[w]_{jj} = 1$ and in general $0 < w_{jk} \leq 1$. The final score is given as:

$$CM = \frac{t(Y, Z) - t(Y, X_{\{NSR\}})}{t(Y, Y) - t(Y, X_{\{NSR\}})} \tag{4.13}$$

where $X_{\{NSR\}}$ is a prediction set where all predictions are the normal class $\{NSR\}$. This metric, which is a weighted version of accuracy [106], is limited to being used on the PhysioNet 2020/21 dataset [3], however, with additional domain knowledge inputs, it can be used in different contexts.

These metrics often fall short in addressing clinical considerations, such as the fact that over-diagnosis is generally less detrimental than missed diagnosis, and they do not sufficiently account for the severity of the diagnosis. In the next section, we will conduct a comprehensive analysis of existing metrics from a clinical standpoint.

4.2.4. ARE ML METRICS CLINICALLY APPLICABLE?

The preceding discussion highlights that a collection of metrics is insufficient for making deployment decisions, underscoring the necessity for a single metric with pertinent attributes to effectively compare different computational models. In the following discussion, we will see that the metrics borrowed from ML do not adequately address clinical needs. Our benchmark for assessing the clinical relevance of these metrics will be based on the criteria outlined in Section 4.2.2. Specifically, we will evaluate whether a wrong diagnosis (WD) is penalized more heavily than a missed diagnosis (MD), which in turn faces a greater penalty than over-diagnosis

(OD), with the perfect diagnosis (PD) receiving the highest scores, i.e.,

$$\text{score}_{WD} < \text{score}_{MD} < \text{score}_{OD} < \text{score}_{PD} \qquad \text{(Clinical Order)}$$

LABEL-BASED METRICS

It is commonly agreed that example-based metrics are more appropriate for evaluating MLC tasks [56]. Nevertheless, for a comprehensive analysis, we will also examine some widely used label-based metrics.

In the subsequent discussion, we shall consider four hypothetical classifiers and their corresponding output sets $\mathcal{P}_O, \mathcal{P}_M, \mathcal{P}_W$, and $\mathcal{P}$, which only have over, missed, wrong, and perfect diagnoses, respectively (e.g., in $\mathcal{P}_O$ we have $y_i \subsetneq \hat{z}_i, \forall(\hat{z}_i, y_i) \in \mathcal{P}_O$).

♦ **Macro precision, macro recall, and macro $\mathbf{F_1}$**–macro precision and macro recall cannot be used in isolation, as we are free to change one at the expense of the other. However, Macro F_1, which is a macro-average of the harmonic means of precision and recall, is a serviceable metric. Macro F_1 is defined as:

$$MacroF_1(\mathcal{P}) = \frac{1}{P}\sum_{j=1}^{P} F_{1(j)}(\mathcal{P}), \text{ where,}$$
$$F_{1(j)}(\mathcal{P}) = \frac{2 \cdot p_j(\mathcal{P}) \cdot r_j(\mathcal{P})}{p_j(\mathcal{P}) + r_j(\mathcal{P})}$$

Where p_j, r_j is precision and recall for the j^{th} class respectively. Consider the case where $r_j(\mathcal{P}_M) \geq p_j(\mathcal{P}_O)$ (note, $p_j(\mathcal{P}_M) = r_j(\mathcal{P}_O) = 1$). Then we have,

$$\frac{2 \cdot r_j(\mathcal{P}_M)}{1 + r_j(\mathcal{P}_M)} \geq \frac{2 \cdot p_j(\mathcal{P}_O)}{1 + p_j(\mathcal{P}_O)}$$
$$\frac{2 \cdot p_j(\mathcal{P}_M) \cdot r_j(\mathcal{P}_M)}{p_j(\mathcal{P}_M) + r_j(\mathcal{P}_M)} \geq \frac{2 \cdot p_j(\mathcal{P}_O) \cdot r_j(\mathcal{P}_O)}{p_j(\mathcal{P}_O) + r_j(\mathcal{P}_O)}$$
$$F_{1(j)}(\mathcal{P}_M) \geq F_{1(j)}(\mathcal{P}_O)$$

If this holds for all j, we have the exact opposite inequality as desired, and even if it is only true for some j, no guarantees can be made that a system that always misses diagnoses is worse than one that always over-diagnoses.

♦ **Micro precision, micro recall, and micro $\mathbf{F_1}$**–similar to their macro counterparts, micro precision and recall cannot be used in isolation, but micro F_1 can be used

independently to evaluate the quality of a computational diagnostic system. It is defined as:

$$MicroF_1(\mathcal{P}) = \frac{2 \cdot \text{micro-precision} \cdot \text{micro-recall}}{\text{micro-precision} + \text{micro-recall}} \quad \text{, where,}$$

$$MicroPrecision(\mathcal{P}) = \frac{\sum_{j=1}^{P} tp_j}{\sum_{j=1}^{P} tp_j + \sum_{j=1}^{P} fp_j} \quad \text{, and}$$

$$MicroRecall(\mathcal{P}) = \frac{\sum_{j=1}^{P} tp_j}{\sum_{j=1}^{P} tp_j + \sum_{j=1}^{P} fn_j}$$

We know $fp_j = 0$ in $\mathcal{P}_M$ and $fn_j = 0$ in $\mathcal{P}_O$. So,

$$\begin{aligned} &MicroPrecision(\mathcal{P}_M) = 1 \\ &\Rightarrow MicroF_1(\mathcal{P}_M) = \frac{2 \cdot MicroRecall(\mathcal{P}_M)}{1 + MicroRecall(\mathcal{P}_M)}, \text{ and, similarly,} \\ &MicroRecall(\mathcal{P}_O) = 1 \\ &\Rightarrow MicroF_1(\mathcal{P}_O) = \frac{2 \cdot MicroPrecision(\mathcal{P}_O)}{1 + MicroPrecision(\mathcal{P}_O)} \end{aligned}$$

So we have $MicroF_1(\mathcal{P}_M) \geq MicroF_1(\mathcal{P}_O)$ whenever, $MicroRecall(\mathcal{P}_M) \geq MicroPrecision(\mathcal{P}_O)$. This means if two diagnostic systems have the same number of true positives and one has a higher number of false positives than the other has false negatives, then $MicroF_1(\mathcal{P}_M) \geq MicroF_1(\mathcal{P}_O)$. This is the opposite of the desired ordering in clinical practice, as false negatives are generally more deleterious.

EXAMPLE-BASED METRICS

In the ensuing discussion, we consider predictions m_i, o_i, and w_i, which are missed, over, and wrong diagnoses, respectively, for the ground truth label y_i, and check if Clinical Order holds. (Note: $m_i \subsetneq y_i, y_i \subsetneq o_i$, and $y_i \cap w_i = \phi$).

♦ **Hamming Loss** is defined as

$$hloss(\mathcal{P}) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{P} |\hat{z}_i \Delta y_i|$$

So, from the definition, we have:

$$hloss(\{(m_i, y_i)\}) = hloss(\{(o_i, y_i)\}) \text{ whenever } |y_i - m_i| = |o_i - y_i|$$

So, missing k diagnoses is penalized just as harshly as producing k over-diagnoses. Since classifiers are tuned to target certain metrics, it must be noted that Hamming loss is usually not optimal for sensitive systems [56].

♦ **Accuracy** is widely known to be an unreliable measure in a clinical context, where imbalanced datasets are the norm [151]. It is defined as

$$\begin{aligned} accuracy(\mathcal{P}) = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{z}_i \cap y_i|}{|\hat{z}_i \cup y_i|}, \text{ so, if we have } |m_i| \cdot |o_i| \geq |y_i|^2 \\ \Rightarrow accuracy(\{(m_i, y_i)\}) \geq accuracy(\{o_i, y_i\}) \end{aligned}$$

Thus, Clinical Order doesn't hold in general. As an example, consider $|y_i| = k \geq 2, |m_i| = k-1, |o_i| = k+2$, then $accuracy(\{(m_i, y_i)\}) = \frac{k-1}{k} \geq \frac{k}{k+2} = accuracy(\{(o_i, y_i)\})$.

♦ **Subset accuracy** is the strictest metric and is defined as

$$\begin{aligned} SAccuracy(\mathcal{P}) &= \frac{1}{N}\sum_{i=1}^{N} I(\hat{z}_i = y_i), \text{ so, we have,} \\ SAccuracy(\{(m_i, y_i)\}) &= SAccuracy(\{(w_i, y_i)\}) \\ &= SAccuracy(\{(o_i, y_i)\}) = 0 \end{aligned}$$

which violates Clinical Order.

♦ **$\mathbf{F_1}$ score** is defined as

$$F_1(\mathcal{P}) = \frac{1}{N}\sum_{i=1}^{N}\frac{2 \times |\hat{z}_i \cap y_i|}{|\hat{z}_i| + |y_i|}$$

Suppose $|y_i| = k, |m_i| = k-1$ (one diagnosis missed), and $|o_i| = k + r$ (r extra predictions). We have:

$$F_1(\{(m_i, y_i)\}) \geq F_1(\{(o_i, y_i)\}) \text{ whenever } r \geq \left\lceil \frac{k}{k-1} \right\rceil$$

So, Clinical Order doesn't hold in general. As in the case of label-based metrics, example based precision and recall aren't meaningful in isolation and aren't discussed here.

♦ **PhysioNet 2020/21 Challenge Metric** is defined in Equations 4.12 and 4.13. Since

w_{jk} is integral to the metric, it is limited for use on the PhysioNet 2020/21 Dataset. Without the weight matrix (i.e., $w = I_{n\times n}$) this is the same as accuracy and inherits all its problems. Even on the PhysioNet 2020/21 dataset, it does not guarantee satisfaction of the inequality (Clinical Order). Of note are the issues introduced by their normalization scheme (as defined in 4.13). Consider the scenario where the ground truth label contains $y_i = \{NSR, a_j, a_k\}$ (NSR is the normal class), and we predict $\hat{z}_1 = \{NSR\}$, and $\hat{z}_2 = \{a_j\}$. We have:

$$
\begin{aligned}
t(y, \hat{z}_1) = t(y, \hat{z}_{\{NSR\}}) &= \frac{1 + w_{j,NSR} + w_{k,NSR}}{3} \\
t(y, \hat{z}_2) &= \frac{1 + w_{NSR,j} + w_{k,j}}{3} \\
CM(y, \hat{z}_1) &= 0 \\
CM(y, \hat{z}_2) &= \frac{1 + w_{NSR,j} + w_{k,j} - 1 - w_{j,NSR} - w_{k,NSR}}{3(t(y,y) - t(y, \hat{z}_1))} \\
CM(y, \hat{z}_2) &= \frac{w_{k,j} - w_{k,NSR}}{3(t(y,y) - t(y, \hat{z}_1))} \\
\Rightarrow CM(y, \hat{z}_2) &< 0, \text{ for some choice of } a_j \text{ (As } w_{jk} \text{ is symmetric.)}
\end{aligned}
$$

Therefore, this metric discourages detection of cardiovascular conditions, in favor of detecting the normal class, which is contrary to clinical expectations.

4.2.5. MEDTRIC – A DOMAIN OBEDIENCE METRIC

In the previous section, we illustrated that most commonly used metrics do not align well with clinical practice. In this section, we propose a new metric designed to meet the established criteria and incorporate clinically desirable properties.

DEFINITION

Given $\mathcal{P}$, consider an instance of prediction and label $\hat{z}_i, y_i$. There are three sets of interest, $\hat{z}_i \cap y_i$, $y_i - \hat{z}_i$, and, $\hat{z}_i - y_i$, corresponding to correct predictions, missed predictions, and extra predictions, respectively (see Fig 4.6). Although $y_i - \hat{z}_i$ and $\hat{z}_i - y_i$ both consist of errors, the former generally has worse clinical outcomes.

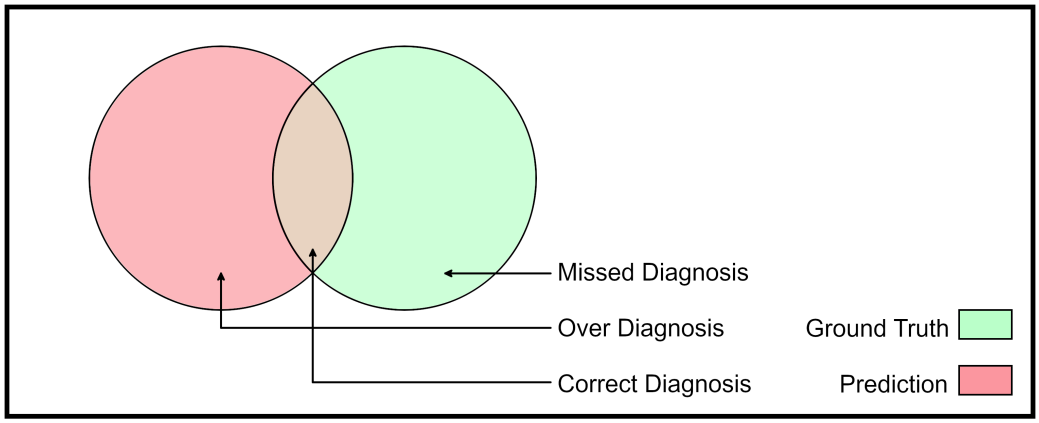


FIGURE 4.6: **The partitions of interest for clinical evaluation.** Figure reproduced from Saha et al. [148].

Since each category poses a unique clinical scenario, we score them as follows:

$$[a_i]_j = \begin{cases} \frac{s_j}{n_j} & \text{if } c_j \in \hat{z}_i \cap y_i \\ \frac{-s_j}{n_j} & \text{if } c_j \in y_i - \hat{z}_i \\ \frac{s_j}{n^*}\left[\frac{1}{|y_i|}\left(\sum_{c_k \in y_i} w_{jk}\right) - 1\right] & \text{if } c_j \in \hat{z}_i - y_i \\ 0 & \text{otherwise} \end{cases} \tag{4.14}$$

where n_i is the number of occurrences of diagnostic condition i in the dataset $\mathcal{P}$. This normalization ensures that the prevalence of diagnostic conditions doesn't affect the final scores. n^* is defined as follows:

$$n^* = \max\{n_j | \forall c_j \in y_i\} \tag{4.15}$$

$\{s_j | \forall j \in \{1 \dots P\}\}$ are significance weights. This reflects the fact that all diagnostic conditions might not be equally relevant, and classes that are critical have a higher value of s_i, so their contribution to the final score is larger. They can all be set to 1 if their relative importance is the same.

w_{jk} measures similarity of diagnostic conditions (as in Alday et al. [3]). This gives partial rewards to over-diagnosis which are of similar nature in outcomes or treatment. If such a matrix is unavailable or not required, w_{jk} can be set to 0 $j \neq k$, and $w_{jj} = 1 \, \forall j$.

If, for a given dataset having P conditions, all 2^P diagnoses are not possible, and contradictory pairs exist (hypo- and hypertension, for instance), we can in-

troduce an additional contradiction penalty term and a contradiction matrix C_{jk}, such that $\mathrm{C}_{jk} = 1$ if condition c_j and c_k can't occur together ($\forall$ patient x, x has $c_j \Rightarrow$ x does not have c_k).

$$[b_i]_j = \begin{cases} \frac{-1}{n_j} \sum_{\forall k \text{ s.t. } c_k \in \hat{z}_i} s_k \cdot C_{jk} \text{ if } c_j \in \hat{z}_i \\ 0, \text{ otherwise} \end{cases} \tag{4.16}$$

Then we can compute the score for the i^{th} instance as follows:

$$t_i = \sum_{j=1}^{P} [a_i]_j + [b_i]_j \tag{4.17}$$

Finally, we sum the scores over all instances in the dataset and normalize. Consider $\mathrm{Y}, \mathrm{Z} = \{y_i | \forall i \in \{1, \ldots, N\}\}, \{\hat{z}_i | \forall i \in \{1, \ldots, N\}\}$, we have $t(\mathrm{Y}, \mathrm{Z})$ defined as follows:

$$t(\mathrm{Y}, \mathrm{Z}) = \sum_{i=1}^{N} t_i \tag{4.18}$$

$$M_{med} = \frac{t(\mathrm{Y}, \mathrm{Z}) - t(\mathrm{Y}, \Phi)}{t(\mathrm{Y}, \mathrm{Y}) - t(\mathrm{Y}, \Phi)} \tag{4.19}$$

Here Φ represents the null prediction set, i.e., $\hat{z}_i = \phi, \forall i \in \{1 \ldots N\}$. This normalization ensures that a perfect classifier gets a maximum possible score of 1, and an inactive one that predicts nothing gets a score of 0. The metric defined in Equation 4.19 is named **MedTric**, which is a portmanteau of **Med**ical Me**tric**.

MEDTRIC FOLLOWS CLINICAL ORDER

CLAIM 4.1 *MedTric always penalizes missed predictions more severely than extra predictions.*

Proof. Since we have the following inequalities;

$$0 < [w]_{jk} < 1 \qquad \forall [w]_{jk}, j \neq k$$

$$0 < \frac{1}{|y_i|} \sum_{c_k \in y_i} [w]_{jk} < 1$$

$$-\frac{s_j}{n_j} < \frac{s_j}{n^*} \left[\frac{1}{|y_i|} \Big(\sum_{c_k \in y_i} [w]_{jk} \Big) - 1 \right] < 0$$

$$(n^* \geq n_j \, \forall j \in \{1 \ldots P\}, \text{ by definition})$$

missed predictions always have heavier penalties than extra predictions.

This does not demonstrate that MedTric follows Clinical Order, and since such a demonstration would be dependent on the exact clinical requirements and details about the dataset, we resort to empirical means in order to validate that Clinical Order is maintained by MedTric. However, MedTric does have desirable behavior in most cases of practical interest. Consider (as in Section 4.2.4) 4 classifiers and their output sets $\mathcal{P}_O, \mathcal{P}_M, \mathcal{P}_W,$ and $\mathcal{P}$, corresponding to over, missed, wrong, and perfect diagnoses respectively, and a specific diagnostic condition a_k.

In $\mathcal{P}_M$ since only missed diagnoses are allowed, we have $fp_k = 0, \hat{z}_i - y_i = \phi$ and the assigned score is given by $\frac{s_k}{n_k}(tp_k - fn_k)$ where tp_k, fp_k, fn_k are the number of true positives, false positives, and false negatives, respectively, for the condition a_k in $\mathcal{P}_M$.

Similarly, in $\mathcal{P}_O$ since only over-diagnoses are allowed, we have $fn'_k = 0, y_i - \hat{z}_i = \phi, tp'_k = n_k$, and the assigned score is given by:

$$
\begin{aligned}
& s_k + \sum_{a_k \notin y_i} \frac{s_k}{n^*}\Big[\frac{1}{|y_i|}\Big(\sum_{c_j \in y_i} w_{kj}\Big) - 1\Big] \\
& \geq s_k + \sum_{a_k \notin y_i} \frac{s_k}{n^*}\Big[\frac{1}{|y_i|}\Big(\sum_{c_j \in y_i} w^*_k\Big) - 1\Big] \\
& = s_k + \sum_{a_k \notin y_i} \frac{s_k}{n^*}\Big[\Big(w^*_k\Big) - 1\Big] \\
& = s_k + fp'_k \frac{s_k}{n^*}\Big(w^*_k - 1\Big) \\
\text{Where } w^*_k &= \min(w_{kj}\ \forall j \in \{1, \ldots, P\})
\end{aligned}
$$

Where, tp'_k, fp'_k, fn'_k are the number of true positives, false positives, and false negatives, respectively, for the condition a_k in $\mathcal{P}_O$. Consider,

$$
\begin{aligned}
\xi_k &= \overbrace{\frac{s_k}{n_k}(tp_k - fn_k)}^{\text{Missed diagnosis score}} - \overbrace{s_k - fp'_k \frac{s_k}{n^*}\Big(w^*_k - 1\Big)}^{\text{Over-diagnosis score}} \\
&= \frac{s_k}{n_k}\Big[tp_k - fn_k - n_k - fp'_k \frac{n_k}{n^*}\Big(w^*_k - 1\Big)\Big] \text{ (as, } tp_k + fn_k = n_k) \\
&= \frac{s_k}{n_k}\Big[fp'_k \frac{n_k}{n^*}\Big(1 - w^*_k\Big) - 2 \cdot fn_k\Big]
\end{aligned}
$$

Now, if $\xi_k < 0 \; \forall k \in \{1, \ldots, P\}$, MedTric follows Clinical Order. Even conservatively, since we have $\frac{n_k}{n^*} \leq 1$ and $0 < 1 - w_k^* < 1$ by definition, $\xi_k < 0$ holds whenever the number of false positives of each condition does not exceed twice the number of false negatives.

If a broader region of operation is required, w_k^* can be adjusted accordingly, e.g., if $w_k^* = \frac{1}{3} \forall k$, MedTric follows Clinical Order whenever the number of false positives of each condition does not exceed thrice the number of false negatives. In more realistic scenarios, however, where prevalence is imbalanced, the region where Clinical Order holds is much broader. For example, if a certain diagnostic condition is a tenth as likely as the most frequent one, we have $\xi_k < 0$ whenever the number of false positives for the condition is less than 20 times the number of false negatives for that same condition.

For $\mathcal{P}_W$, we have $tp_k = 0, fn_k = n_k, \hat{z}_i \cap y_i = \phi$, and the score corresponding to a_k is given by $-s_k - fp'_k \frac{s_k}{n^*}\left(1 - w_k^*\right) \leq -s_k$, which is the lowest possible missed diagnosis score.

Thus, depending on the clinical context and its associated tolerance for missed diagnosis vs. over-diagnosis, we can choose the values of w_{jk} such that MedTric is guaranteed to follow Clinical Order (see example in Table 4.8).

TABLE 4.8: **Example of scoring for missed, over and wrong diagnoses.**
O, M, W, P stands for over, missed, wrong and perfect diagnoses, respectively. The following subscript number represents the quantity, e.g., O_1 means one over-diagnosis. *MedTric* sorts them in the desired clinical order (labels are drawn from PhysioNet dataset).

Ground Truth	**Prediction**	**Type**	**Score**
CRBBB, AF, QAb	LAD, STach, TInv	W_3	-0.230
CRBBB, AF, QAb	LAD, STach	W_2	-0.159
CRBBB, AF, QAb	LAD	W_1	-0.081
CRBBB, AF, QAb	ϕ	M_3	0.0
CRBBB, AF, QAb	CRBBB	M_2	0.25
CRBBB, AF, QAb	CRBBB, AF	M_1	0.75
CRBBB, AF, QAb	CRBBB, AF, QAb, LAD, NSIVCB	O_2	0.756
CRBBB, AF, QAb	CRBBB, AF, QAb, LAD	O_1	0.918
CRBBB, AF, QAb	CRBBB, AF, QAb	P	1

TABLE 4.9: **Example illustrating dataset prevalence independence.**
Here in the two cases shown above, the underlying classification quality is the same; conditions A and B are detected 100% of the time, and condition X is detected 50% of the time; only the prevalence in the dataset has changed (in Case 1, $\{X, A\}$ occurs 10% of the time and in Case 2, 90% of the time). However, unlike other metrics (e.g., F1 score), this doesn't change the MedTric score, thus demonstrating dataset prevalence invariance.

	Prediction$\rightarrow$	$\{X, A\}$	$\{A\}$	$\{A, B\}$	**Total**	$\mathbf{F_1}$	**MedTric**
Case 1	**GT:** $\{X, A\}$	50	50	0	100	0.983	0.833
	GT: $\{A, B\}$	0	0	900	900		
Case 2	**GT:** $\{X, A\}$	450	450	0	900	0.850	0.833
	GT: $\{A, B\}$	0	0	100	100		

DATASET ARTIFACTS

If a computational system is X% accurate in one diagnostic class and Y% in another, some metrics may change solely due to variations in the proportions of these classes. Micro-averaged label-based metrics and example-based metrics are particularly vulnerable to this issue. Evaluating model performance can be obscured by demographic artifacts, especially given the common issue of class imbalance in diagnostic datasets [179].

To address this problem, we suggest normalizing each score contribution by the corresponding class frequency (see Equations 4.14 and 4.16), ensuring that the final score is independent of dataset proportions and represents the true per-instance accuracy (refer to Table 4.9).

Our proposal aligns well with the principles of cost-sensitive learning [39]. As discussed in the preceding section, our metric imposes a greater penalty for false negatives (missed diagnoses) than for false positives (over-diagnosis). Furthermore, we have disentangled the prevalence of diagnostic conditions from performance measures, recognizing that rarity does not necessarily equate to severity.

Beyond prevalence, diagnostic datasets often incorporate a notion of criticality, which is not typically reflected in standard machine-learning metrics. This aspect of criticality necessitates an additional layer of cost-based decision-making. The significance weights s_j (see Equations 4.14 and 4.16) ensure that classifiers that underperform on critical classes face harsher penalties. These weights are normalized so they can be interpreted as the *contribution of a particular diagnostic class to the*

final score.

Additionally, by introducing w_{jk} and C_{jk}, we account for interactions between different diagnostic classes in a manner that is informed by domain knowledge. For instance, if two diagnostic conditions have similar prognoses or treatment plans, misclassifying one as the other might be penalized less severely [3]. This framework can be interpreted as a cost-sensitive learning problem with a cost matrix $\mathrm{M} \in \mathbb{R}^{2^P \times 2^P}$ where every misclassification of a set $\alpha \in 2^A$ as another set $\beta \in 2^A$ carries a potentially distinct cost.

4.2.6. MEDTRIC IN PRACTICE

In the previous section, we demonstrated that our metric ensures compliance with Clinical Order (i.e., ensuring the scoring aligns with the clinical severity monotonic order) under certain conditions. We also asserted that MedTric maintains this property in most practical scenarios. Our analysis suggests that other relevant metrics fail to adhere to Clinical Order, often in common situations. To facilitate a fair comparison, in this section, we will evaluate each metric against a consistent set of diagnostic scenarios to assess their suitability in a clinical context.

Metric scores often rely on the frequency of various classes within the evaluation dataset, potentially obscuring performance deficiencies in specific classes due to their rarity. Conversely, our metric is designed to ensure score invariance with changes in the prevalence of diagnostic conditions. We aim to investigate how often these conditions are violated (if at all) by the various metrics under consideration. As computational diagnostic systems, particularly those employing machine learning methods, are tuned to particular metrics, inconsistencies between these metrics and clinical practice can lead to model behaviors that are similarly misaligned.

DATASETS AND IMPLEMENTATION DETAILS

In order to measure these, we use three publicly available multi-label diagnostic datasets from different diagnostic disciplines and modalities. The first is the PhysioNet dataset, which is described in Section 4.1.4. The weight matrix used for computation of CM and M_{med} is borrowed from the work by Alday et al. [3], and the contradiction matrix $[C]_{ij}$ equals 1 when the pair a_i, a_j cannot occur simultaneously, and is 0 otherwise (see Table 4.1). The significance values were determined by breaking down the possible diagnoses into three groups, namely supercritical, critical, and noncritical. A weight of 1 was assigned to supercritical, 0.8 to critical

and 0.6 for noncritical conditions. Their constituents are given in Table 4.10.

TABLE 4.10: **Significance weights for different diagnoses.**
1 is assigned to super critical group, 0.8 to critical group and 0.6 to non-critical group.

Supercritical	**Critical**	**Non-critical**	
LQRSV	SA	STach	NSIVCB
TAb	Brady	PR	IRBBB
AF	LQT	PVC	PAC
AFL	IAVB	SVPB	RBBB
LBBB	SB	LPR	LAD
CRBBB	QAb	VPB	NSR
	LAnFB	RAD	

The second is the CheXpert [79] dataset, which contains 224,316 chest radiographs of 65,240 patients, labeled with 14 classes of findings from frontal and lateral X-rays. They have uncertainty labels, along with positive and negative labels for all 14 classes.[3] Finally, we employed a multi-label free text classification dataset[132] consisting of 978 samples labeled with 45 ICD-9 codes. For both these datasets, we used $[s]_i = 1\ \forall i$, $[w]_{ij} = \frac{2}{3}\ \forall i, j, i \neq j$, and $[w]_{ii} = 1\ \forall i$. $[C]_{ij}$ was taken to be 0.

4.2.7. EXPERIMENTS AND RESULTS

CLINICAL ORDER

In order to check violations of monotonicity (Clinical Order), we first sample a data point (x_i, y_i) from the dataset $\mathcal{D}$ in question. Following this, we generate Γ candidate predictions $\hat{z}_{i\gamma}$ such that:

$$f_{\text{random}}(y_i) = \hat{z}_{i\gamma} = \{a_m | P(a_m \in \hat{z}_{i\gamma} | a_m \in y_i) = p, \text{ and, } P(a_n \in \hat{z}_{i\gamma} | a_n \notin y_i) = 1 - q\} \tag{4.20}$$

This simple model f_{random} emulates a classifier that has a sensitivity of p and specificity of q in each class. Next we group the predictions into several buckets, each with a particular type of diagnosis (wrong, missed, over, or perfect) and the degree

[3] All uncertain labels were assumed to be false as per the zeros strategy in Irvin et al. [79].

(count) of the same.

$$\begin{aligned}
&\overbrace{\hat{z}_{i(0)}, \hat{z}_{i(1)}, \ldots,}^{t_w \text{ wrong}} \overbrace{\hat{z}_{i(c_1)}, \ldots,}^{t_w-1 \text{ wrong}} \overbrace{\hat{z}_{i(c_2)}, \ldots,}^{\ldots} \overbrace{\hat{z}_{i(c_3)}, \ldots, \hat{z}_{i(c_4)},}^{1 \text{ wrong}} \\
&\overbrace{\hat{z}_{i(c_4+1)}, \ldots,}^{t_m \text{ missed}} \overbrace{\hat{z}_{i(c_5)}, \ldots,}^{t_m-1 \text{ missed}} \overbrace{\hat{z}_{i(c_6)}, \ldots,}^{\ldots} \overbrace{\hat{z}_{i(c_6)}, \ldots, \hat{z}_{i(c_7)},}^{1 \text{ missed}} \\
&\overbrace{\hat{z}_{i(c_6+1)}, \ldots,}^{t_o \text{ over}} \overbrace{\hat{z}_{i(c_7)}, \ldots,}^{t_o-1 \text{ over}} \overbrace{\hat{z}_{i(c_8)}, \ldots,}^{\ldots} \overbrace{\hat{z}_{i(c_9)}, \ldots, \hat{z}_{i(c_{10})},}^{1 \text{ over}} \overbrace{\ldots, \hat{z}_{i(k)}}^{\text{perfect}}
\end{aligned} \tag{4.21}$$

Then, we compute the metric score $\mathcal{M}\big(\{y_i\}, \{f_{\text{random}}(y_i)\}\big)$ for each candidate group, and check if monotonicity is followed, i.e.,

$$\begin{aligned}
\mathcal{M}(y_i, W_i^{t_w}) &< \mathcal{M}(y_i, W_i^{t_w-1}) < \ldots < W_i^1) \\
&< \mathcal{M}(y_i, M_i^{t_m}) < \mathcal{M}(y_i, M_i^{t_m-1}) < \ldots < M_i^1) \\
&< \mathcal{M}(y_i, O_i^{t_o}) < \mathcal{M}(y_i, O_i^{t_o-1}) < \ldots < \mathcal{M}(y_i, O_i^1) < \mathcal{M}(y_i, P)
\end{aligned} \tag{4.22}$$

Where $W_i^t = \{\hat{z}_{i(\gamma)} | \hat{z}_{i(\gamma)}$ has t wrong diagnosis$\}$ (and similarly for O, M). Then we repeat this with several (ρ) samples (x,y) from the dataset to estimate the probability (τ) that metric $\mathcal{M}$ follows clinically applicable monotonic order.

We used $\rho = 100$ samples from the datasets to probe each metric for monotonicity with 4 pairs of (p, q) and repeated the experiment $n = 10$ times to gather statistics. Unsurprisingly, only MedTric obeys monotonicity 100% of the time (see Figure 4.7). Note that subset accuracy and Hamming loss never obey expected clinical ordering, thus making them least suited for evaluation of diagnostic systems.

PREVALENCE INVARIANCE

Promising computational techniques often depend heavily on large amounts of data, yet handling long-tailed datasets remains a considerable challenge. Consequently, if a diagnostic system excels in one class but underperforms in another-—particularly when instances from the poorly performing class are rare-—certain metrics might not effectively highlight this weakness (see Table 4.9). Given that imbalanced datasets are a common reality in diagnostics, it is crucial for metrics to detect these potential blind spots accurately.

To test for this property, we select two classes from the dataset a_M and a_m which are the most and least frequently occurring classes, respectively. Then we create a

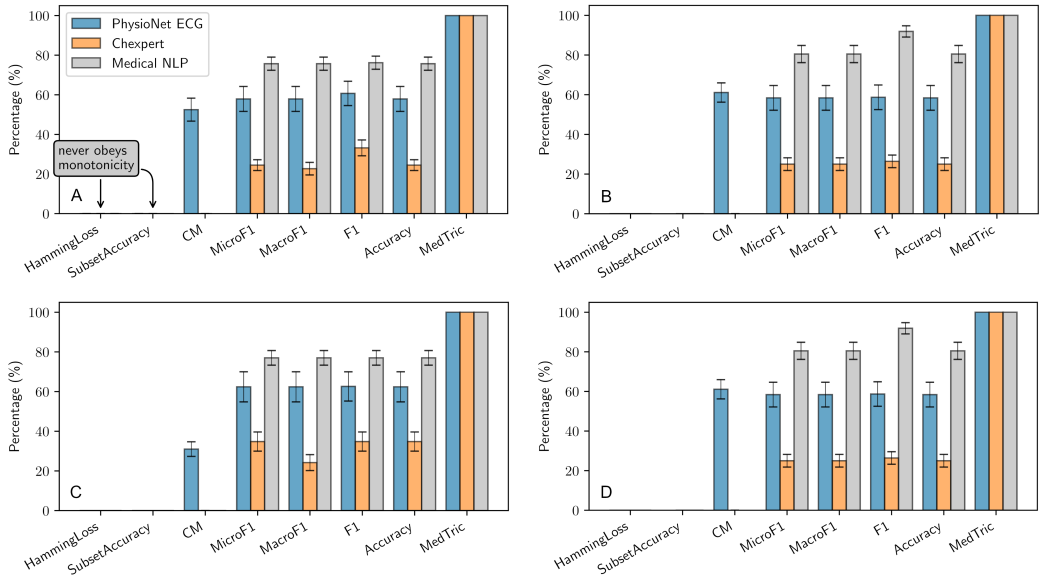


FIGURE 4.7: **MedTric is the only metric maintaining clinically applicable order 100% of the time.**
The X-axis displays the metric under evaluation, and the Y-axis shows the percentage of times monotonicity is followed by a particular metric. The experiment is carried out with 4 sensitivity and specificity settings $A-(80\%,95\%)$,$B-(80\%,90\%)$, $C-(60\%,95\%)$, $D-(60\%,90\%)$ over the three datasets. Hamming loss and subset accuracy never follow monotonicity. CM was only computed on PhysioNet dataset. Figure is reproduced from Saha et al. [148].

subset $\mathcal{D}_\alpha \subset \mathcal{D}$ of such that

$$\mathcal{D}_\alpha = \{\hat{z}_i | P(a_M \in \hat{z}_i) = \alpha, \text{ and, } P(a_m \in \hat{z}_i) = 1-\alpha \; \forall i \in \{1,\ldots,l\}\} \quad (4.23)$$

Thus the dataset contains roughly $l\alpha$ instances of class a_M and $l(1-\alpha)$ instances of a_m. Next, we generate predictions $g_{\text{random}}(\mathcal{D}_\alpha)$ based on $\mathcal{D}_\alpha$ following the protocol outlined in the previous section with sensitivity p_M, p_m for a_M, a_m respectively (and specificity q). The quantity we are interested in estimating is the standard deviation of the metric, i.e., $\sigma(\mathcal{M})$, which will measure the amount of variation it has when subjected to variations in the dataset. This is given as:

$$\sigma(\mathcal{M}) = \left(\mathop{\mathbb{E}}_{\alpha \sim U(0,1)} \left[\mathcal{M}\big(\mathcal{D}_\alpha, g(\mathcal{D}_\alpha)\big)^2 \right] - \mathop{\mathbb{E}}_{\alpha \sim U(0,1)} \left[\mathcal{M}\big(\mathcal{D}_\alpha, g(\mathcal{D}_\alpha)\big) \right]^2 \right)^{\frac{1}{2}} \quad (4.24)$$

We estimate this quantity with a Monte-Carlo simulation by drawing η samples from $U(0,1)$. For our experiments, we set $p_M = 0.9$ ("good" performance for the

abundant class), and $p_m = 0.5$ ("poor" performance for the rare class). $\eta = 50$ samples were drawn for $\alpha \sim U(0, 1)$, and for each α, $l = 100$ samples for a_M and a_m were drawn to create $\mathcal{D}_\alpha$. The experiment was repeated $n = 10$ times each, for $q = 99\%, 95\%$, and over all three datasets (see Figure 4.8).

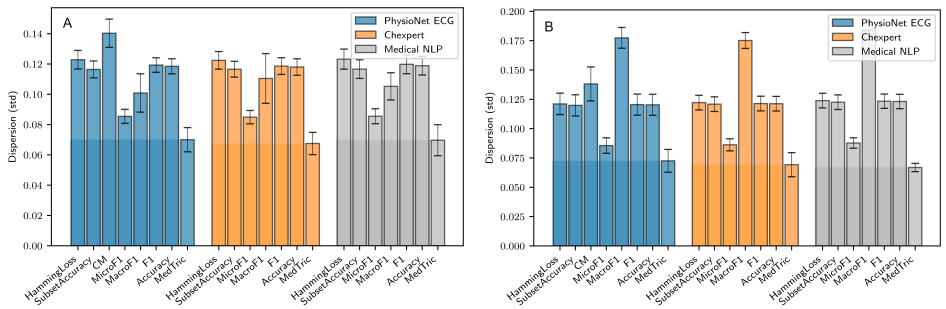


FIGURE 4.8: **Dispersion (σ) of various metrics with change in dataset prevalence** A - $q = 95\%$ and B - $q = 99\%$. Metric scores are often dictated by the frequency of occurrence of certain diagnostic conditions in the evaluation dataset and are not indicative of the actual performance of the computational diagnostic system. High dispersion scores indicate that a metric is likely to obscure weaknesses of diagnostic systems due to relative prevalence of classes. MedTric outperforms other metrics in this regard. Figure is reproduced from Saha et al. [148].

We consistently observe (see Figure 4.8) that our metric has the least dispersion and therefore is most likely to capture weaknesses of diagnostic systems that would otherwise be obfuscated by rarity.

4.3 DISCUSSIONS

In Section 4.1, we further explore the inability of DL systems to adhere to domain rules, a problem exacerbated by noisy annotations. To combat this, we introduce the **DOST** algorithm, which incorporates logical constraint information into DL systems via self-supervision. We also present empirical studies demonstrating that the **DOST** paradigm not only diminishes rule violations, thereby enhancing the alignment of models with domain rules, but also results in performance improvements. This is accomplished by repurposing data that might otherwise have been discarded as noisy. Given that all other variables remain constant across the different training strategies, we infer that these performance enhancements are likely attributable to the integration of domain knowledge into the models.

In Section 4.2, we showed that current metrics for evaluating multi-label computa-

tional diagnostics often fail to capture the intricacies of clinical practice adequately. Specifically, commonly used bipartition task metrics do not effectively address the risks associated with missed diagnoses, over-diagnoses, and incorrect diagnoses in a clinically sound manner. We have also demonstrated that metric outcomes can be obscured by prevalence, leading to an inaccurate reflection of actual performance. When transplanting ML metrics to a clinical setting, important clinical features, such as the relative importance of different diagnoses and appropriate penalties for unrealistic predictions, have been largely overlooked.

Our metric, however, addresses these key clinical requirements, aligning more closely with clinical practice. It preserves the order relation between different types of diagnostic errors in terms of real-world consequences. Furthermore, it handles contradictions and clinical significance, rewarding models in a manner consistent with diagnostic practice. This metric allows for straightforward comparison of computational models tackling the same problem, even when calculated over datasets with differing diagnostic distributions. Even though MedTric was designed with a clinical settings in mind, it can be adapted to any MLC problem where domain constraints can inform the ranking of the various types of errors that a system might make.

Constraint-aware learning and evaluation are equally important considerations for the deployment of DL systems in critical scenarios. Since models are often tuned to maximize performance according to a target metric, promulgation of constraint-obedience metrics would favor adoption of constraint-obedient predictive systems. Similarly, development of constraint-aware learning strategies such as **DOST**, necessitate investigation into constraint-adherence metrics for an accurate assessment of deployability. Thus, further research into these areas serve to complement and bolster each other.

5

CONSTRAINED INFERENCE

[1]Modern LLMs demonstrate remarkable skills in a plethora of language tasks like reasoning, coding, wordplay, QA, etc. [117, 196]. However, their ability to generate language in a constrained setting is somewhat under-explored. The ability to control or direct the generation process in order to meet certain criteria is extremely desirable in several realms of NLP, but it remains a challenge [134].

These constraints can take several forms, like length limitations, character limitations, rhyming/meter-based constraints, and restriction to formal languages. As a simple example, consider the task of poem generation [134], where in addition to thematic aptness, we must contend with a rhyming scheme. Further, for certain kinds of poems, like sonnets or haikus, we have extended impediments on the number of syllables or characters. Similar constraints arise for generation of lyrics where a certain pattern of stressed and unstressed syllables might be desirable based on the beats of music.

Constraints also frequently arise when dealing with formal languages like those defined by a finite automaton or a context-free grammar (CFG) [93]. These problems have become imperative in the current LLM zeitgeist, since in addition to coding, LLMs have to interact with databases, application programming interfaces (APIs), tools, interpreters, etc. In such use cases, generations must follow strictly laid-out

[1]This chapter is largely based on our paper titled *"Language Models are Crossword Solvers"* [147].

guidelines to be useful. There is also a growing body of literature studying *LLMs as agents* [110] or *embodied LLMs* [189], and in these use cases, language generation must follow physical and environmental constraints as well.

In this chapter, we study *crossword puzzles*, which are a form of word game typically played using a square grid of black and white squares. The goal of the puzzle is to fill in the white squares with letters based on provided clues (see Figure 5.1). In addition to strict limitations on the number of admissible letters for each answer, crossword puzzles feature further constraints in the form of letter interactions of various clues.

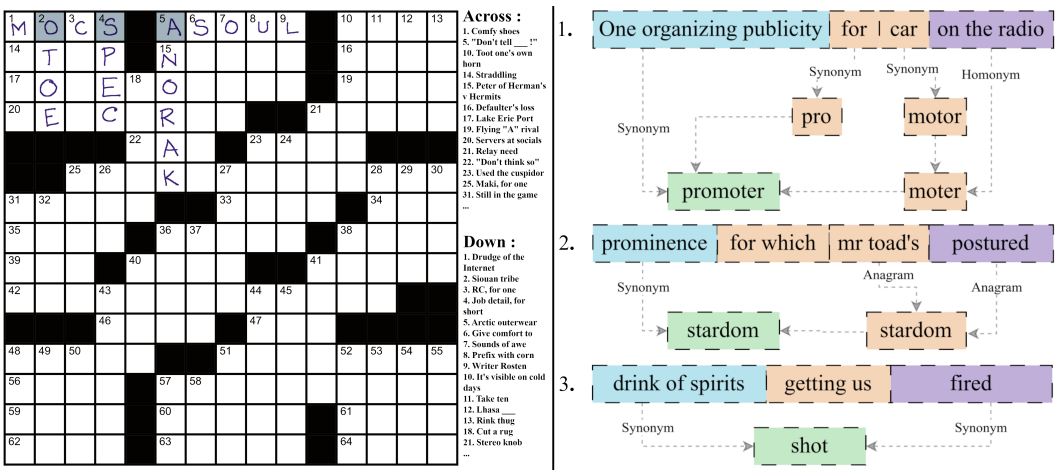


FIGURE 5.1: **Example of a crossword puzzle (left) and cryptic clues (right).**
(*left*) The grid must be filled up with answers from the semantic clues provided. The gray highlighted squares produce additional constraints, e.g., first character of the answer to clue 1 (across) and clue 4 (down) must be the same. Example by Fred Piscop.
(*right*) In cryptic crosswords, the clues involve some form of wordplay and synonyms and often involve world knowledge. Examples are taken from the cryptonite [36] dataset. Figure is reproduced from Saha et al. [147].

In this chapter, we first discuss necessary background for crosswords and some previous attempts at automated crossword solving. We then analyze LLM's abilities at this task and present an algorithm that can solve crosswords with the aid of LLMs. We finally present experimental results validating the performance of our algorithm and further studies exploring different aspects of LLM performance with regard to generalizability, reasoning, and abilities at different linguistic tasks.

5.1 BACKGROUND

Successfully solving crosswords demands a high level of mastery in interpreting contextual clues, semantics, wordplay, character manipulation, world knowledge, arithmetics, and reasoning (see Figure 5.1). Additionally, it requires adhering to constraints such as length restrictions and character overlaps. For wider proliferation of LLMs, it is essential for them to exhibit the ability to comply with constraints that may arise from knowledge graphs, formal languages, tabular data, or other domain-specific demands. Hence, examining crossword solving can improve the adaptability of LLMs to relatively less explored domains where constraints coexist with linguistic challenges. Given that crossword solving encompasses several desirable competencies, and identifying areas for improvement could benefit other linguistic applications, the aim of this chapter is to evaluate the abilities of LLMs in tackling this multifaceted task.

Clue: Laser-focused mindset (12)
Answer: TUNNELVISION

(a) 'TUNNELVISION' is a synonym for 'Laser-focused mindset'

Clue: Vegas nickname (7)
Answer: SINCITY

(b) 'Vegas', a shorthand for Las Vegas, Nevada, is also known as 'Sin City'.

FIGURE 5.2: **Examples of straight crossword clues with explanations.** Examples are taken from xwordinfo.com.

Two kinds of crosswords are studied in this chapter: the first is the American style, or *straight* crossword, which typically features denser grids with relatively straightforward clues, and solutions typically involve synonyms or world knowledge (see Figure 5.2 for examples). The other form of crossword studied is the UK style, or *cryptic* crossword. These have sparser grids but feature more formidable clues involving anagrams, homophones, world knowledge, puns, string manipulation, and a variety of domain-specific tactics. An example *cryptic* clue and solution is outlined in Figure 5.3 and further examples can be found in Figure 5.1.

Traditional methodologies for solving *straight* crossword puzzles comprise two primary elements: a candidate answer proposal system and a grid-filling algorithm [6, 55]. The candidate answer proposal systems often employ similarity-based search on massive clue-answer databases, fine-tuned LMs, or a combination of both [164,

```
Clue: Culminating point of story about Judy's husband by the railway track
(5,4)
Answer: PUNCH LINE
```

FIGURE 5.3: **Example of cryptic crossword clue with explanation.**
Answering this clue requires connecting "Judy" to the popular puppet show *Punch and Judy*, and inferring that "`Judy's husband`" refers to "Punch". Additionally, we must observe that "railway track" is synonymous with "line", and combining these gives "`PUNCH LINE`" which also means "Culminating point of story". Example is taken from lovattspuzzles.com.

187]. Grid-filling, on the other hand, utilizes versions of constraint satisfaction problem (CSP) algorithms. An example is the system *Proverb* [104], which achieves a 98.1% letter accuracy on New York Times (NYT) crosswords. Wallace et al. [187] fine-tuned BERT and ByT5 on a dataset of 6.4 million clue-answer pairs and, using a belief propagation algorithm, reached a 99.7% letter accuracy. Kulshreshtha et al. [96] set benchmarks using foundational language models, underscoring this task as "*... a new high bar for AI systems*". They attempted to solve NYT crossword puzzles using foundational LMs in conjunction with a Satisfiability Modulo Theory (SMT) solver, achieving limited success, and ultimately had to prune the crossword grid based on candidate generations and ground truth answers. Our focus in this work is to examine the capability of foundational LLMs at this task, rather than improving upon automated crossword-solving systems.

Cryptic crosswords pose a greater challenge, and conventional algorithms using large datasets of clues along with a CFG parser [27] have performed poorly, achieving only 7% accuracy [146]. Recent research has explored leveraging LLMs to solve cryptic crossword clues. Efrat et al. [36] compiled an extensive dataset of cryptic crossword clues from UK newspapers such as *The Times* and *The Telegraph* and fine-tuned a T5-Large [135] model to establish baseline performances. They utilized a training split ensuring mutual exclusivity between the training and test sets to prevent memorization. Rozner et al. [141] gathered a dataset from *The Guardian* and fine-tuned a T5-Large model through curriculum learning, demonstrating performance enhancements. They criticized Efrat et al. [36]'s methodology, arguing that a disjoint train-test split does not sufficiently teach models to solve cryptic crosswords, as models show "... robustness to plural and other inflections." Instead, they suggested grouping similar-root words in a split and found that this more rigorous criterion led to decreased performance. Sadallah et al. [146] analyze the performance of

contemporary LLMs like Mistral-7B [82], LLaMA2-7B [180], and ChatGPT [125] in few-shot settings, in addition to fine-tuning the Mistral model. They found that ChatGPT outperformed other models, achieving an accuracy of 9.5%. They noted several limitations in their study, such as the restricted set of LLMs used and the potential for data contamination.

These recent studies on cryptic crossword solving with LLMs highlight a *significant performance gap between LLMs and human experts*, who solve 99% of cryptic crossword clues [146]. However, these studies approach the problem as a QA task and *neglect the constraints imposed by the grid*. Further investigation is needed to explore whether an appropriate method that incorporates constraint information into LLMs can lead to substantial performance improvements.

DATASETS

Our analysis primarily utilizes three crossword puzzle datasets. The first two datasets, *Cryptonite*[2] by Efrat et al. [36] and *word-init-disjoint*[3] (abbreviated as *Init*) by Rozner et al. [141], pertain to *cryptic* crossword puzzles. The methodological differences between *Cryptonite* and *Init*, as discussed in the preceding section, are not significant to our work since we *do not perform any training*. We randomly selected 2000 samples for reporting results, and in-context examples were also randomly drawn from a substantial pool of samples that did not overlap with the testing set.

TABLE 5.1: **Details of various crossword datasets used.**
Results are primarily reported using the NYT *(straight)*, Cryptonite, and Init *(cryptic)* datasets. Further, some smaller datasets are used to test for generalizability and reasoning. The NYT (Grids) dataset refers to a dataset of 100 full crossword puzzles we collected and contains 7700 clues alongside grid information like their position, orientation, etc.

Dataset	**Train**	**Validation**	**Test**
Cryptonite	470,804	26,156	26,157
word-init-disjoint	75,847	32,628	33,905
NYT (Clues)	10,000		2,000
NYT (Grids)	*(test only)* 100 grids ≡ 7700 clues		
After May 20, 2024 – *(post-cutoff)*			
Lovatts			242
The Guardian			200

[2] github.com/aviaefrat/cryptonite
[3] github.com/jsrozner/decrypt

Finally, we collected a dataset from the *New York Times*[4] for our analysis of *straight* crossword puzzles. In addition to 12,000 randomly sampled clue-answer pairs split into two sets–*test* (2000) and *support* (10,000)–we collected 100 randomly sampled Monday crossword puzzles ranging from 20th January 1969 to 7th August 2023 with all clues (7700) and grid information, which are used to report results for full grid solving. All three sets are completely disjoint. Most of the results are presented on the *NYT* dataset for *straight* crosswords and *Init* for cryptic crosswords, as it posed a greater challenge for LLMs compared to *Cryptonite*.

Some additional "post-cutoff" datasets,[5] i.e., puzzles published after the training cut-off time of every LLM considered in this study, were collected for testing generalization performance and analysis of reasoning abilities. Dataset statistics are summarized in Table 5.1.

MODELS

TABLE 5.2: **Models used in this study and their details.**
The open weights models were run on a server consisting of 2× 80G A100 NVIDIA GPUs and were implemented in `PyTorch` [130] and `huggingface` [199]. All models were used in `bf16` format whenever supported. The proprietary models were used with their respective APIs.

Model	Params.	Context Length	Knowledge Cut-off
Phi 3 mini Instruct [1]	3.8B	4K	Oct. 2023
Mistral v0.2 Instruct [82]	7B	32K	Dec. 2023
LLaMA 3 Instruct [116]	8B	8K	Mar. 2023
Mixtral v0.1 [83]	8x7B	32K	Dec. 2023
LLaMA 2 [180]	70B	4K	Sep. 2022
LLaMA 3 [116]	70B	8K	Dec. 2023
Claude 3 sonnet `20140229` [7]	?	200K	Mar. 2024
GPT-3.5-Turbo-`0125` [126]	?	16K	Sep. 2021
GPT-4-Turbo `2024-04-09` [126]	?	128K	May 2024

We employed various open-source and proprietary LLMs in our study with varying architectures (mixture of experts, grouped query attention, etc.) and a wide range of parameter scales. The details of the model are summarized in Table 5.2. Our generations were achieved with the random sampling strategy (with $\texttt{T} = 0.5$) in all cases, and all other settings were left unchanged. For few-shot prompt responses, `max-tokens` were set to 10, and for chain-of-thought [197] responses, `max-tokens`

[4] nytimes.com/crosswords
[5] theguardian.com, lovattspuzzles.com

were set to 1000.

5.2 PRELIMINARY EXPERIMENTS

In this section we present some preliminary experiments involving various crossword subtasks like clue-answering, constraint information incorporation, counting, etc.

5.2.1. CROSSWORD CLUE SOLVING

Following in the paradigm of Kulshreshtha et al. [96], Rozner et al. [141], Sadallah et al. [146], and Efrat et al. [36], we first analyze this problem through the lens of crossword clue answering. Typically, the first step in crossword solving is generating candidate answers based on clues, in a QA fashion, where the clue and the associated length serve as the question, and we expect the LLM to come up with the answer. We test several LLMs with the prompt in Figure 5.4.

```
[{ "role":"system",
    "content": "You are an expert crossword solver. Given a clue
        please provide the best possible answer succinctly. Do
        not produce extra text.\n The number of characters in the
        answer is given in brackets and must be strictly adhered
        to. e.g. Clue: Daily update (4)// means the answer should
        have 4 characters."}{
   "role":"user",
    "content": "Clue: <1st in-context example> (length) // <answer>\n
                Clue: < ...  > (length) // answer 2\n
                Clue: < ...  > (length) // answer k\n
                Clue: <query clue> (length) // "
}]
```

FIGURE 5.4: Few-shot prompt for crossword clue solving.

We used 5 and 10 in-context examples[6] and reported results with Phi 3 3.8B Instruct [1], Mistral 7B Instruct [82], Llama 2 70B [180], Llama 3 8B Instruct, Llama 3 70B [116], Mixtral 8x7B [83], Claude 3 Sonnet [7], GPT 3.5 Turbo, and GPT 4 Turbo [126] to cover a wide range of parameter scales and a mix of open-weights and proprietary models. The results are summarized in Figure 5.5.

[6]Further increases (25-shot) did not yield performance benefits.

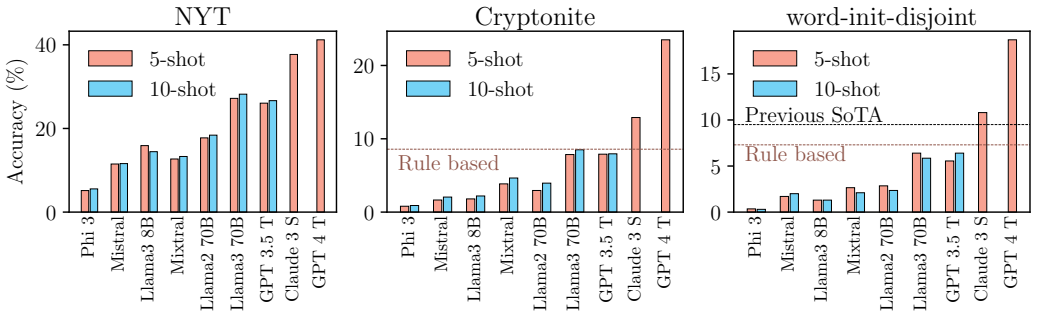


FIGURE 5.5: **Analyzing LLMs' ability to generate answers from crossword clues.** We test LLMs at different scales on the NYT, Cryptonite, and *Init* datasets with 5-shot and 10-shot prompts. All results are with `T=0.5`, and the figure is reproduced from Saha et al. [147].

Notably, 5- and 10-shot prompts[7] yielded similar performance results, and there is a significant performance difference (up to 5%) in all models between *Cryptonite* and *Init*, with LLMs showing diminished performance on the *Init* dataset.

The analysis of model performance across a range of datasets reveals improved outcomes with increased model scale. On the NYT dataset, we register accuracies (exact match) of 27.2% for Llama 3 70B, 26.05% for GPT 3.5 Turbo, 37.7% for Claude 3 Sonnet, and **41.2%** for GPT-4-Turbo. While large language models underperform on *cryptic* crosswords compared to *straight* crosswords, Claude 3 Sonnet and GPT-4-Turbo manage to surpass previous SoTA results in *cryptic* crossword datasets. Their accuracies are recorded at 12.9% and **23.5%** on Cryptonite, and 10.8% and **18.7%** on *Init*, respectively. Notably, GPT-4-Turbo achieves a significant **1.97**× improvement over the previous SoTA result, which reported an accuracy of 9.5% as reported by Sadallah et al. [146] (see Figure 5.5).

5.2.2. CHARACTER CONSTRAINT ADHERENCE

During the course of tackling a crossword puzzle, solvers often deal with partly-filled grids, where solutions to some clues have already been unearthed. These intermediate states provide valuable hints that solvers strategically use to guide their choices for the remaining answers. For instance, consider Figure 5.1: when attempting to solve the clue in position 14 (across), one can leverage the letters already determined from position 2 (down) and 4 (down). This narrows the possibilities for the answer to those that match the pattern "`_ T _ P`". In this section, we examine the capabilities

[7] We did not perform experiments with 10-shot prompts on Claude and GPT-4-Turbo due to budget limitations.

of LLMs in utilizing such constraints to improve their responses. Our prompting strategy for this task is similar to the QA task in the preceding section, but we additionally provide *"letter masks"*, which are supposed to serve as constraining information in order to guide generation. When creating a query for a particular test instance with $k\%$ hints, we randomly selected N few-shot instances and ensured the few-shot examples also had $k\%$ hints. The number of characters revealed (h) is given by the formula:

$$h = \max\left(1, \text{round}\left(\frac{k}{100} \times len(\text{answer})\right)\right) \forall k > 0 \tag{5.1}$$

h many characters are randomly selected and revealed, all other characters are replaced with "_". The prompt structure is given in Figure 5.6.

```
[{ "role":"user",
    "content": "Clue: <clue 1> (3) // _ _ N => MEN\n
               Clue: < ...  > (6) // _ _ E _ _ _ => BREATH\n
               Clue: <query clue> (length) // _ _ X _ X => "
}]
```

FIGURE 5.6: Prompt used to solve crossword clues with character hints.

For this experiment, we selected the top-performing open weights and proprietary models, specifically LLaMA 3 70B and GPT-4-Turbo. Additionally, we included results from smaller models to assess whether the observed trends persist across model sizes. The evaluation was conducted on the NYT and *Init* datasets using 5-shot prompts. In each query, $k\%$ of the answer's characters are provided alongside the clue and the expected length of the answer. The LLMs are tasked with "unmasking" the remaining characters by leveraging the given constraints and the crossword clue. Our results are summarized in Table 5.3.

As reflected in Table 5.3, for both datasets under consideration, we observe that in nearly every scenario, LLMs demonstrate enhanced performance as the percentage of constraint information increases. Furthermore, to benchmark GPT-4-Turbo against the previously reported SoTA results by Sadallah et al. [146], we conducted our experiment using their same settings and dataset split. The findings reveal that *GPT-4-Turbo*, with 5-shot prompts, achieves an accuracy of **76.3%**, significantly surpassing the fine-tuned Mistral 7B model's **27%** accuracy by a **factor of 2.8**. This ability of LLMs to effectively utilize constraints for deciphering crossword clues

TABLE 5.3: **Can LLMs exploit character constraints from a partially filled grid?** Sadallah et al. [146] reported an accuracy of 27.0% (70% hinted clues) by fine-tuning a Mistral 7B model on the *Init* dataset, *which GPT-4-Turbo* (**76.30%** accuracy) *outperforms by a factor of* ∼**2.8**× *without fine-tuning*. All results are with 5-shot prompts.

Hint (%)	**0%**		**25%**		**50%**		**70%**
	NYT	*init*	NYT	*init*	NYT	*init*	*init*
Mistral 7B	10.95%	1.70%	9.70%	2.80%	11.95%	4.80%	
LlaMa 3 8B	15.8%	1.30%	19.7%	2.85%	24.65%	6.25%	
LlaMa 3 70B	27.20%	6.40%	31.80%	11.45%	45.30%	20.35%	
GPT 4 Turbo	41.2%	18.70%	59.95%	33.70%	75.75%	52.85%	**76.30%**

indicates that they are well-equipped for the comprehensive task of solving entire crosswords.

5.2.3. LENGTH CONSTRAINT ADHERENCE

Despite notable advancements in performance, SoTA LLMs struggle with adherence to length constraints, which suggests a difficulty with counting characters within words or phrases-—which we refer to as *sub-token counting*. Even the highest-performing model, GPT-4-Turbo, generates responses of incorrect length on 26.2% and 16.9% of the *Init* and the NYT datasets, respectively. This challenge may be attributed to the tokenization technique employed by LLMs, such as Byte-Pair Encoding [159]. In the transformer model architecture [183], the first layer converts character tokens to embedding vectors, leading to a loss of information about the individual characters. This character-level detail must be recovered during training. While the exact mechanism by which LLMs reacquire this information remains unclear, it is plausible that they learn from training data that explicitly include length specifications.

There are websites[8] that offer extensive lists of words along with their respective lengths. Often replies on message boards include a count of the number of characters in a piece of text. Artifacts like these, which contain sufficient information to infer token lengths, go on to become part of the datasets that LLMs are trained on. We propose that LLMs learn to count sub-tokens based on this information provided during training.

To explore this hypothesis further, we developed a sub-token counting task where the

[8] word.tips/words-by-length for example.

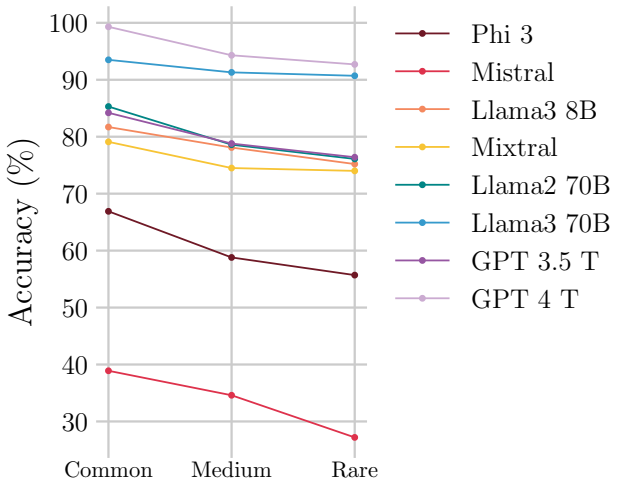


FIGURE 5.7: **Can LLMs count?**
LLMs ability to count the number of characters in a word declines with the unigram frequency of the word, suggesting that counting is somewhat familiarity-based. Figure reproduced from Saha et al. [147].

LLM is provided with a sequence of lowercase characters without spaces and tasked with predicting the total number of characters in the sequence. For the purpose of testing our hypothesis, we selected three sets of 1,000 English words each, named *Common*, *Medium*, and *Rare*, based on the word unigram frequencies assembled by Segaran and Hammerbacher [158] from Google's Trillion Words corpus. The ranks for these sets are 1 - 5,000 for *Common*, 47,500 - 52,500 for *Medium*, and 95,000 - 100,000 for *Rare* words.

If language models possess a broadly generalizable ability to perform sub-token counting, we would expect their counting accuracy to remain consistent across words of varying prevalence. However, our observations suggest otherwise. As illustrated in Figure 5.7, the accuracy of LLMs in the sub-token counting task diminishes as the frequency of the token decreases, a trend evident across all tested models.

To understand whether sub-token counting performance differs between words included in the model's vocabulary and randomly generated gibberish with identical length distributions, we conducted a further experiment.[9] We first assembled a set of words by intersecting the vocabulary of every open-source model under consideration with the list of the top 100,000 words, ensuring that these "words" are extremely likely[10] to be vocabulary tokens across all evaluated models and are not special

[9] It is possible that the performance dip observed is because rare words are generally longer than common words.

[10] This cannot be confirmed for proprietary models.

tokens (like <bos>, for example). Next, we generated a set of gibberish words by substituting each character in the vocabulary set words with a randomly selected character from the set {a-z}, thereby ensuring that both word sets have identical length distributions.

TABLE 5.4: LLM counting performance for vocabulary words and gibberish.

Model	Vocab. [Acc. (%)]	Gibberish [Acc. (%)]
Phi 3 3.8B Instruct	79.4	61.2
Mistral 7B Instruct	47.9	28.2
Llama 3 8B Instruct	92.6	69.7
Mixtral 8x7B	92.6	80.1
Llama 2 70B	92.8	80.0
Llama 3 70B	99.6	87.5
GPT 3.5 Turbo	86.0	62.1
GPT 4 Turbo	99.8	98.8

Our findings (see Table 5.4) indicate that counting accuracy not only varies with token frequency but also shows a significant disparity between the accuracy for *vocabulary* words vs. *gibberish* words. Although these results do not definitively prove that LLMs depend on memorized instances from their training data to execute sub-token counting, they provide compelling evidence suggesting LLMs may indeed learn to count from artifacts in the training data that contain length information.

5.3 LLM-GUIDED SEARCH

In this section, we attempt to address the challenge of completing crossword grids with the assistance of LLMs. This problem requires more than merely generating correct answers for the given clues; it also involves capitalizing on constraints set by words already placed on the grid. Additionally, it necessitates backtracking to revise previous candidates that may no longer be valid with the emergence of new information. Given that LLMs exhibit the capability to exploit partially filled grids, when coupled with an appropriate search algorithm, they can potentially be effective in solving crosswords.

Our proposed algorithm - *SweepClip*, initiates by generating a set of candidate answers for all the clues in the crossword (*Sweep*) and removing any answers that do not fit using a graph based criterion (*Clip*). Following this, constraints derived

from the previously accepted answers are utilized to generate neighboring candidate answers[11] and further prune candidates that do not fit correctly. This strategy is applied iteratively until one of the following conditions is met: (i) the entire crossword is successfully filled, (ii) the number of iterations surpasses a predetermined limit (`max_iter`), or (iii) the LLM computational budget is exhausted. Our algorithm implicitly performs self-consistency checks to improve candidate answers; e.g., an answer generated at the first sweep may be discarded only to be accepted in later iterations, when it is consistent with a larger number of other answers.

Formally, a crossword puzzle consists of a grid and a set of clues $C = \{c_1, \ldots c_n\}$ and answers (ground truth) $A = \{a_1, \ldots a_n\}$ corresponding to position i in the grid. The grid imposes a graph $G = (V, E)$, where $V = \{v_1, \ldots v_n\}$ is the set of vertices corresponding to every clue/answer in the crossword, and:

$$E = \{(v_i, v_j) |\ \forall i, j\ i \neq j\ s.t.\ a_i, a_j \text{ share a grid position}\} \tag{5.2}$$

Given an LLM, and a set of clues $C' \subseteq C, C' = \{c_{j_1}, c_{j_2}, \ldots\}$ corresponding to vertices $\{v_{j_1}, v_{j_2}, \ldots\}$ we can generate candidate answers $\hat{A} = \{\hat{a}_{j_1}, \hat{a}_{j_2}, \ldots\}$, where $\hat{a} = LLM(c)$. We abbreviate this as $\hat{A} = LLM(C')$. For a subset V' of V, let the set of clues associated with V be denoted by $C(V')$.

Two candidate answers $\hat{a}_i, \hat{a}_j$ are said to be in **conflict**, if μ-th position of a_i and ν-th position of a_j are in the same grid position, however $\hat{a}_i[\mu] \neq \hat{a}_j[\nu]$, i.e., the μ-th character of a_i and ν-th character of a_j are different.

There are two sub-graphs $G_p(\hat{A})$ and $G_n(\hat{A})$ of G that correspond to a set of candidate answers $\hat{A}$. Let v_i, v_j correspond to $\hat{a}_i, \hat{a}_j \in \hat{A}$. An edge $(v_i, v_j) \in E$ is in $G_p(\hat{A})$ if and only if $\hat{a}_i, \hat{a}_j$ don't conflict, else it is in $G_n(\hat{A})$. We call the largest connected component of a graph H, $LCC(H)$, and for a subset S of vertices V of a graph G, $ngbd(G, S)$ denotes the vertices in V that are adjacent to S but not in S. The algorithm is detailed in Algorithm 2.

For pruning, we employ the largest connected component from the answers generated thus far, thereby ensuring a degree of coherence among the selected answers. While this approach is somewhat restrictive—since isolated answers can indeed be correct—it proves effective in eliminating incorrect answers at an early stage. This prevents the propagation of errors by avoiding the use of potentially incorrect con-

[11] Using prompts with appropriate letter masks, see Figure 5.6.

Algorithm 2 : SweepClip

Given C, crossword graph G and an LLM.
Generate $\hat{A} \leftarrow LLM(C)$.
for i in $\{1, \ldots,$ `max_iter`$\}$ **do**
    Construct $G_p(\hat{A}), G_n(\hat{A})$
    $L \leftarrow LCC(G_p(\hat{A}))$.
    $\hat{A} \leftarrow \{\hat{a}_i |\ v_i \in L\}$
    while $G_n(\hat{A})$ has edges. **do**
        A max degree vertex in $G_n(\hat{A}) \rightarrow v_m$
        Remove v_m from $G_n(\hat{A})$.
        $\hat{A} \leftarrow \hat{A} - \hat{a}_m$
    end while
    Calculate $N \leftarrow ngbd(G, \hat{A})$.
    Augment $C(N)$ with character information.
    $\hat{A} \leftarrow \hat{A} \cup LLM(C(N))$
    if `solved` or `budget_exceeded` **then**
        `break`
    end if
end for

straints that could lead to further incorrect answers in subsequent iterations.

5.4 EXPERIMENTS

In Section 5.2, we performed some preliminary experiments to analyze the capabilities of LLMs in the QA clue answering task and sub-token counting. Although the QA results are promising, it is possible to further improve performance with prompting techniques like *chain-of-thought* [197] and *self-consistency* [193]. Our results are summarized in Table 5.5 alongside previously reported SoTA results. Our best result on the *word-init-disjoint* split is **20.85%**, which *improves over the previous SoTA (9.5%) by a factor of* **2.2**$\times$ *without any fine-tuning*.

In the following sections we present performance validation results of our *SweepClip* algorithm and further experiments testing generalizability and reasoning abilities of SoTA LLMs.

5.4.1. FULL CROSSWORD SOLVING

To report performance results of our algorithm *SweepClip* (Algorithm 2), we use the set of 100 randomly sampled Monday *NYT* crossword puzzles (see Table 5.2).

TABLE 5.5: **Comparison of our results with previously reported SoTA results.** Results are on the *Init* dataset with crossword clue deciphering treated as QA. SFT refers to supervised fine-tuning and CoT(1)@3SC refers to *Chain-of-thought* [197] prompting (1 shot) with self-consistency [193] (3 samples).

Model	**Method**	**Accuracy** EM (%)
Rule-based [27]	CFG+WordNet	7.3
T5 [36]	SFT	1.1
T5 [141]	Curriculum Learning	6.5
Mistral 7B [146]	SFT	1.2
Mistral 7B [146]	10 shot	4.6
Chat GPT [146]	3 shot	9.5
GPT-4-Turbo **(ours)**	5 shot	18.70
GPT-4-Turbo **(ours)**	CoT(1)@3SC	**20.85**

Monday crosswords are typically easier, and we restrict ourselves to them solely to limit computational cost. We used two LLMs for this: GPT-4-Turbo and Llama 3 70B. The results are produced with a `max_iter` of 30 and a budget of 0.5 US dollars per crossword for GPT-4-Turbo and a `max_iter` of 35 and a budget of 600 LLM calls for LLaMA 3 70B.

TABLE 5.6: Results from solving NYT crosswords with *SweepClip*.

Error Tolerance	**% of Crosswords**	
	LLaMa 3	GPT-4 T
100% solved	0	**48**
$\leq$ 1 character error	1	**55**
$\leq$ 5 character error	10	**71**
$\geq$ 90% Accuracy	30	**80**
$\geq$ 50% Accuracy	82	**98**

Our experiments (see Table 5.6) demonstrate that *SweepClip* paired with GPT-4-Turbo is able to solve **48%** of crosswords without any errors and **55%** of crosswords with at most 1 wrong character. The average character level accuracy in crossword solving is **93.1%** ($\pm$ 14.1%). Our algorithm improves the clue-wise answer accuracy[12] (exact match) to **89.6%** ($\pm$ 16.9%) from the base accuracy (without the algorithm), which is 43.5% ($\pm$ 23.5%), an improvement of **2.1**$\times$. The previously

[12]Clue level accuracy is different from character level accuracy [96]; e.g., it is possible to have a filled-in crossword without deciphering all clues.

reported SoTA accuracy on this task with a foundational LLM (without fine-tuning) was **26%** with retrieval-augmented generation and an SMT solver coupled with an oracle that eliminates parts of the crossword grid that do not have suitable generated answers [96].

When we apply our algorithm with the smaller LLaMA 3 70B model, the overall performance does degrade; however, the final clue-answering accuracy still sees an uplift, reaching **59.4%** ($\pm$ 24.1%) compared to a baseline of just 22.3% ($\pm$ 14.4%). Thus, through the application of our algorithm, we've exploited constraint information to enhance LLM performance, significantly overtaking QA techniques typically employed for clue deciphering. To the best of our knowledge, *this is the first demonstration of an algorithm that successfully solves crosswords utilizing out-of-the-box LLMs*.

5.4.2. GENERALIZABILITY

DATA CONTAMINATION

We observed significant performance gains in SoTA LLMs across the board, which might point to potential data contamination, i.e., the models have seen some of the clue-answer pairs during training. To ensure that our observed performance enhancements were not merely a result of data contamination, we assembled additional datasets consisting of cryptic crossword clues, all sourced from puzzles released after *May 20, 2024, which is after the knowledge cutoff for all LLMs evaluated* (see Table 5.1). To guard against any inadvertent duplications, we checked the answers in these *post-cutoff* datasets against the entire pool of cryptic crossword datasets utilized within our study, totaling 665,497 answers. We found no duplicates in the *post-cutoff* Guardian set and only two in the *post-cutoff* Lovatts set, which were removed. The *Init* dataset is also derived from *The Guardian*, ensuring that the results displayed in Table 5.7 maintain consistency.

Since *no appreciable difference in performance on the post-cutoff dataset* (see Table 5.7) is observed, we conclude that these LLMs can generalize beyond potential contamination in their training set.

HUMAN EVALUATION

To determine the capacity of models to reason about cryptic crossword clues, we perform **human evaluation**. We used a 3-shot *Chain-of-thought* prompt to elicit a

TABLE 5.7: **Performance of LLMs on *post-cutoff* datasets.**
Note, *Init* by Rozner et al. [141], also sourced their data from *The Guardian*, thus these results provide a fair head-to-head comparison of performance. We report exact match (%).

Model	**Lovatts**	**Guardian**	**Init**
Llama 3 70B	26.03%	5.5 %	6.4 %
Claude 3 Sonnet	46.28%	12.5%	10.8%
GPT 4 Turbo	61.57%	18.5%	18.7%

reasoned response to crossword clues from GPT-4-Turbo, which are then analyzed for soundness vis-à-vis factual and logical errors. In checking whether a justification for an answer given by the LLM is logically and factually sound, we assess grammatical soundness and phraseological meaningfulness of the sentences in the answer, the existence of counterfactual statements (e.g., "BULKY has 4 characters", "the initial letters of ARE RATS TIRED NOW are ARTS", etc.), and whether it presents a statement as an inference from previous statements when it does not follow from those, etc. If an answer by the LLM is found unsatisfactory in any of these aforementioned areas, it is labeled **unsound**. All responses were evaluated by 3 annotators[13] and in case of conflicting answers (4 out of 100, Fleiss' $\kappa = 0.94$), discussions were held to reach a consensus. We chose the 100 samples from the post-cutoff *Lovatts* set for this to allay concerns of contamination.

TABLE 5.8: **Results from human evaluation.**
An answer is called **correct** if the model prediction exactly matches the ground truth. The answer is called sound if it contains no logical or factual errors. Results are with GPT-4-Turbo on the *post-cutoff* Lovatts set.

	Sound (61)	**¬Sound** (39)
Correct (65)	48%	17%
Wrong (35)	13%	22%

The results (see Table 5.8) show that **74%** of the time GPT-4-Turbo provided a correct answer, it also gave *sound reasoning in support of the answer*. This leads us to conjecture that they possibly have a significant ability to reason and generalize.

[13] authors of Saha et al. [147].

POTENTIAL PITFALLS

To further analyze if LLMs demonstrate any common failure modes, we manually tagged the human evaluation dataset based on the principal skill required to solve a particular puzzle clue. The skill-based categories are:

- **ANG**–the answer is an *anagram* of some words of the clue (e.g., `Cubit is mixed up cookie` → `BISCUIT`).
- **HOM**–the answer is a *homophone* of some words of the clue (e.g., `Heard prints are for royalty` → `PRINCE`).
- **CNT**–the answer is disguised in a contiguous section of the clue (e.g., `The Press leaves pres enter s to go in` → `ENTER`).
- **SCJ**–the answer is found by combining several words that are synonyms of various parts of the clue (e.g., `Reasonable food allowance for Capone` → `RATION AL`).
- **OTH**–this class lumps together a variety of other skills like Spoonerisms, acronyms, world knowledge, and various other kinds of character manipulations (e.g., `Pi p er loses heart on jetty` → `PIER`).

TABLE 5.9: **Are there common failure modes for LLMs?**
ANG refers to *anagrams*, **SCJ** refers to *synonym conjugation*, **CNT** refers to *containment*, **HOM** refers to *homophones*, and **OTH** refers to *others*. The percentages in parentheses refer to the prevalence of a particular type of clue in the database.

Lovatts	**ANG** (27%)	**CNT** (26%)	**SCJ** (18%)	**HOM** (8%)	**OTH** (21%)
GPT 4 Turbo	74%	56%	50%	100%	67%

(a) Results for *Lovatts* dataset (human-annotated).

Init+Cryptonite	**ANG** (8.5%)	**CNT** (2.5%)
Llama 3 70B	2.6%	12.2%
Claude 3 Sonnet	7.9%	15.3%
GPT 4 Turbo	25.0%	33.7%

(b) Results for *Init+Cryptonite* dataset.

Due to the limited number of human annotations, our results (Table 5.9a) are not statistically significant; however, these results hint at the fact that GPT-4-Turbo

demonstrates strong performance in *anagrams* and *homophone*-based clues, boasting 74% and 100% accuracy, respectively (baseline[14] - 65%).

We extended this analysis to the *Cryptonite* and *Init* datasets (Table 5.9b), which together include 4000 clue-answer pairs, finding similar trends: GPT-4-Turbo maintained respectable anagram performance with 25% accuracy over a 21.1% baseline. Conversely, Llama 3 70B recorded a meager 2.6% accuracy, lagging behind a 7.16% baseline, and Claude 3 Sonnet mirrored this trend with 7.9% accuracy compared to an 11.85% baseline. These findings suggest that Llama 3 70B and Claude 3 Sonnet may struggle more with anagram-based clues. It should be noted that unlike **(ANG, CNT)**, categories like **(HOM, SCJ, OTH)** cannot be reliably identified automatically, presenting a limitation in broader analysis for these types of clues.

To better understand the influence of *sub-token counting performance* on clue-solving capabilities, we crafted an additional experiment. We consider all such clues where the model successfully interpreted the clue's semantics but failed to comply with the specified length constraints (e.g., `LECTURER` instead of `PROFESSOR` or `NANNA` instead of `GRANNY`). We counted the number of *wrong LLM predictions with high semantic similarity to ground truth answers*.[15] We found that GPT-4-Turbo and Llama 3 70B, respectively, produced length error predictions **46.4%** and **59.9%** of the time. These findings underscore how weak length constraint adherence ability severely hampers the clue-solving potential of LLMs, highlighting a key area for improvement.

■

[14]Baseline refers to mean accuracy across all kinds of clues.
[15]Similarity score of 0.5 or higher as measured by OpenAI `text-embedding-3-large`

6

CONCLUSION

There have been undeniably massive strides in the capabilities of modern deep learning-based systems, as evidenced by their widespread adoption across several fields. However, their inability to leverage established domain-specific principles is inexpedient. Furthermore, their tendency to violate these principles is a major hindrance to their adoption in critical fields like medicine, law, industrial automation, robotics, avionics, etc., where data-driven decision models must operate within domain-specific constraints. An approach to learning that acts within the framework of laid-out rules and leverages these rules alongside data to improve predictive performance is a key objective of deep learning research.

This dissertation first analyzes deep learning-based systems with regard to their ability to adhere to domain constraints when learning from data alone. A persistent sentiment echoed by many researchers probing this area is the dearth of large-scale rule-supported datasets; to address this issue, we put forth a large-scale logical understanding evaluation dataset with rule annotations. This dataset helps us demonstrate that large-scale training with an abundance of data does not guarantee domain obedience. Similarly, intervention-based approaches allow us to show that language models are not semantically faithful, i.e., they do not behave in accordance with domain expectations. Following this, a new technique to improve constraint adherence is presented, and issues with the current paradigm of domain-blind evaluation of models are highlighted. Finally, we explore domain coherence in the frontier of

deep learning-based natural language research–large language models–and propose techniques to improve their domain obedience.

Domain knowledge-aware deep learning promises to be a major milestone towards scientific applications of deep learning. In this vein, we undertook a preliminary study attempting to address a problem arising from cosmology, where the known laws of physics serve as constraining information. Results from this study are presented in Appendix A.

The findings in this dissertation are a step towards a general framework for data-driven learning systems that operate within and learn from domain-specific requirements, and they pave the way for encouraging future research opportunities. In particular, agentic systems that reason toward achieving goals could greatly benefit from domain-knowledge guidance, and further work addressing this challenge would be compelling. Additionally, a natural extension of this work is to couple constraint-aware models with contemporary explainability techniques. Such integration would allow us to assess whether the model-generated rationales align with the underlying domain rules and could facilitate the creation of a transparent and consistent audit trail.

While the focus of this dissertation has been on advancing the applicability of deep learning-based systems in domain-specific contexts, it is also vital to consider the broader societal implications of these technological advancements. Although widespread use of automation technologies holds significant promise, it also threatens the economic futures of large swathes of the population in the current societal landscape. Policy shifts prioritizing strong social safety nets are crucial for mitigating these effects.

PUBLICATIONS

Following is a list of publications pertinent to this dissertation.

- *MedTric : A clinically applicable metric for evaluation of multi-label computational diagnostic systems*. *Journal*
  Soumadeep Saha, Utpal Garain, Arijit Ukil, Arpan Pal, Sundeep Khandelwal.
  PLOS ONE, 18(8), 1–19; 2023. [10.1371/journal.pone.0283895] [148].

- *Analyzing Semantic Faithfulness of Language Models via Input Intervention on Question Answering*. *Journal*
  Akshay Chaturvedi, Swarnadeep Bhar, **Soumadeep Saha**, Utpal Garain, Nicholas Asher.
  Computational Linguistics, 50(1), 119-155; 2024. [10.1162/coli_a_00493] [18].

- *VALUED - Vision and Logical Understanding Evaluation Dataset*. *Journal*
  Soumadeep Saha, Saptarshi Saha, Utpal Garain.
  Journal of Data-centric Machine Learning Research (DMLR), (13):1–18; 2024. MLR press [150].

- *LADDER: Revisiting the Cosmic Distance Ladder with Deep Learning Approaches and Exploring Its Applications*. *Journal*
  Rahul Shah, **Soumadeep Saha**, Purba Mukherjee, Utpal Garain, Supratik Pal.
  The Astrophysical Journal Supplement Series (ApJS), 273(2), 1–27; 2024. The American Astronomical Society. [10.3847/1538-4365/ad5558] [162].

- *Language Models are Crossword Solvers*. *Conference*
  Soumadeep Saha, Sutanoya Chakraborty, Saptarshi Saha, Utpal Garain.
  2025 Annual Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics [NAACL 2025 (main)], 2074–2090; 2025. Association for Computational Linguistics. [10.18653/v1/2025.naacl-long.104] [147].

- *DOST–Domain Obedient Self-supervision for Trustworthy Multi Label Classification with Noisy Labels*. *Workshop*
  Soumadeep Saha, Utpal Garain, Arijit Ukil, Arpan Pal, Sundeep Khandelwal.
  Proceedings of the 8th International Workshop in Health Intelligence, The Association for the Advancement of Artificial Intelligence (AAAI), 2024.
  AI for Health Equity and Fairness: Leveraging AI to Address Social Determinants of Health, 117–127; 2024. Springer Nature Switzerland. [10.1007/978-3-031-63592-2_10] [149].

BIBLIOGRAPHY

[1] Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Qin Cai, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Yen-Chun Chen, Yi-Ling Chen, Parul Chopra, Xiyang Dai, Allie Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Victor Fragoso, Dan Iter, Mei Gao, Min Gao, Jianfeng Gao, Amit Garg, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Mahoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Ce Liu, Mengchen Liu, Weishung Liu, Eric Lin, Zeqi Lin, Chong Luo, Piyush Madan, Matt Mazzola, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Xin Wang, Lijuan Wang, Chunyu Wang, Yu Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Haiping Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Sonali Yadav, Fan Yang, Jianwei Yang, Ziyi Yang, Yifan Yang, Donghan Yu, Lu Yuan, Chengruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *Preprint*, arXiv:2404.14219.

[2] Guillaume Alain and Yoshua Bengio. 2017. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

[3] Erick A Perez Alday, Annie Gu, Amit J Shah, Chad Robichaux, An-Kwok Ian Wong, Chengyu Liu, Feifei Liu, Ali Bahrami Rad, Andoni Elola, Salman Seyedi, Qiao Li, Ashish Sharma, Gari D Clifford, and Matthew A Reyna. 2021. Classification of 12-lead ECGs: the PhysioNet/computing in cardiology challenge 2020. *Physiological Measurement*, 41(12):124003.

[4] Rami Aly, Steffen Remus, and Chris Biemann. 2019. Hierarchical multi-label classification of text with capsule networks. In *Proceedings of the 57th Annual*

Meeting of the Association for Computational Linguistics: Student Research Workshop, pages 323–330, Florence, Italy. Association for Computational Linguistics.

[5] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 39–48. IEEE Computer Society.

[6] Giovanni Angelini, Marco Ernandes, and Marco Gori. 2005. Webcrow: A web-based crosswords solver. In *Intelligent Technologies for Interactive Entertainment*, pages 295–298, Berlin, Heidelberg. Springer Berlin Heidelberg.

[7] Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku.

[8] Rubén Arjona, Hai-Nan Lin, Savvas Nesseris, and Li Tang. 2021. Machine learning forecasts of the cosmic distance duality relation with strongly lensed gravitational wave events. *Phys. Rev. D*, 103(10):103513.

[9] Sriram Balasubramanian, Naman Jain, Gaurav Jindal, Abhijeet Awasthi, and Sunita Sarawagi. 2020. What's in a name? are BERT named entity representations just as good for any other name? In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 205–214, Online. Association for Computational Linguistics.

[10] Solon Barocas, Moritz Hardt, and Arvind Narayanan. 2019. *Fairness and Machine Learning*. fairmlbook.org.

[11] Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

[12] Jose Luis Bernal, Licia Verde, and Adam G. Riess. 2016. The trouble with h_0. *JCAP*, 10:019.

[13] Esben Jannik Bjerrum. 2017. Smiles enumeration as data augmentation for neural network modeling of molecules. *ArXiv preprint*, abs/1703.07076.

[14] Andrea Borghesi, Federico Baldo, and Michela Milano. 2020. Improving deep learning models via constraint-based domain knowledge: a brief survey. *ArXiv preprint*, abs/2005.10691.

[15] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

[16] David Camarena and Valerio Marra. 2020. A new method to build the (inverse) distance ladder. *Mon. Not. Roy. Astron. Soc.*, 495(3):2630–2644.

[17] Sitthichok Chaichulee, Chissanupong Promchai, Tanyamai Kaewkomon, Chanon Kongkamol, Thammasin Ingviya, and Pasuree Sangsupawanich. 2022. Multi-label classification of symptom terms from free-text bilingual adverse drug reaction reports using natural language processing. *PLOS ONE*, 17(8):1–22.

[18] Akshay Chaturvedi, Swarnadeep Bhar, Soumadeep Saha, Utpal Garain, and Nicholas Asher. 2024. Analyzing semantic faithfulness of language models via input intervention on question answering. *Computational Linguistics*, 50(1):119–155.

[19] Zhao-Min Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo. 2019. Multi-label image recognition with graph convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 5177–5186. Computer Vision Foundation / IEEE.

[20] Ethan A. Chi, John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577, Online. Association for Computational Linguistics.

[21] Davide Chicco and Giuseppe Jurman. 2020. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21(1):6.

[22] Blender Online Community. 2018. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam.

[23] Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20.

[24] Antonio J. Cuesta, Licia Verde, Adam Riess, and Raul Jimenez. 2015. Calibrating the cosmic distance scale ladder: the role of the sound horizon scale and the local expansion rate as distance anchors. *Mon. Not. Roy. Astron. Soc.*, 448(4):3463–3471.

[25] G Cybenko. 1989. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2.

[26] Tirtharaj Dash, Sharad Chitlangia, Aditya Ahuja, and Ashwin Srinivasan. 2022. A review of some techniques for inclusion of domain-knowledge into deep neural networks. *Scientific Reports*, 12(1):1040.

[27] Robin Deits. 2015. rdeits/cryptics.

[28] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society.

[29] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

[30] Konstantinos F. Dialektopoulos, Purba Mukherjee, Jackson Levi Said, and Jurgen Mifsud. 2023. Neural network reconstruction of cosmology using the pantheon compilation. *Eur. Phys. J. C*, 83(10):956.

[31] Konstantinos F. Dialektopoulos, Purba Mukherjee, Jackson Levi Said, and Jurgen Mifsud. 2024. Neural network reconstruction of scalar-tensor cosmology. *Phys. Dark Univ.*, 43:101383.

[32] Michelangelo Diligenti, Marco Gori, and Claudio Saccà. 2017. Semantic-based regularization for learning and inference. *Artificial Intelligence*, 244:143–165. Combining Constraint Solving with Mining and Learning.

[33] Emily C Dimmer, Rachael P Huntley, Yasmin Alam-Faruque, Tony Sawford, Claire O'Donovan, Maria J Martin, Benoit Bely, Paul Browne, Wei Mun Chan, Ruth Eberhardt, Michael Gardner, Kati Laiho, Duncan Legge, Michele Magrane, Klemens Pichler, Diego Poggioli, Harminder Sehra, Andrea Auchincloss, Kristian Axelsen, Marie-Claude Blatter, Emmanuel Boutet, Silvia Braconi-Quintaje, Lionel Breuza, Alan Bridge, Elizabeth Coudert, Anne Estreicher, Livia Famiglietti, Serenella Ferro-Rojas, Marc Feuermann, Arnaud Gos, Nadine Gruaz-Gumowski, Ursula Hinz, Chantal Hulo, Janet James, Silvia Jimenez, Florence Jungo, Guillaume Keller, Phillippe Lemercier, Damien Lieberherr, Patrick Masson, Madelaine Moinat, Ivo Pedruzzi, Sylvain Poux, Catherine Rivoire, Bernd Roechert, Michael Schneider, Andre Stutz, Shyamala Sundaram, Michael Tognolli, Lydie Bougueleret, Ghislaine Argoud-Puy, Isabelle Cusin, Paula Duek-Roggli, Ioannis Xenarios, and Rolf Apweiler. 2011. The UniProt-GO annotation database in 2011. *Nucleic Acids Res*, 40(Database issue):D565–70.

[34] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

[35] Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. 2020. Adversarial and domain-aware BERT for cross-domain sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4019–4028, Online. Association for Computational Linguistics.

[36] Avia Efrat, Uri Shaham, Dan Kilman, and Omer Levy. 2021. Cryptonite: A cryptic crossword benchmark for extreme ambiguity in language. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4186–4192, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

[37] Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

[38] Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 20d. Amnesic Probing: Behavioral Explanation with Amnesic Counterfactuals. *ArXiv preprint*, abs/d.

[39] Charles Elkan. 2001. The foundations of cost-sensitive learning. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'01, page 973–978, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[40] Celia Escamilla-Rivera, Maryi Carvajal, Cristian Zamora, and Martin Hendry. 2022. Neural networks and standard cosmography with newly calibrated high redshift grb observations. *JCAP*, 04(04):016.

[41] Andre Esteva, Brett Kuprel, Roberto A. Novoa, Justin Ko, Susan M. Swetter, Helen M. Blau, and Sebastian Thrun. 2017. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115–118.

[42] Richard Evans and Edward Grefenstette. 2018. Learning explanatory rules from noisy data. *J. Artif. Int. Res.*, 61(1):1–64.

[43] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. 2015. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136.

[44] Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

[45] Jun-Peng Fang and Min-Ling Zhang. 2019. Partial multi-label learning via credible label elicitation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3518–3525. AAAI Press.

[46] FIDE. 2023. International chess federation handbook.

[47] Evelyn Fix and J. L. Hodges. 1989. Discriminatory analysis. nonparametric discrimination: Consistency properties. *International Statistical Review / Revue Internationale de Statistique*, 57(3):238–247.

[48] Manoel V. França, Gerson Zaverucha, and Artur S. D'avila Garcez. 2014. Fast relational learning using bottom clause propositionalization with artificial neural networks. *Machine Learning*, 94(1):81–104.

[49] Wendy L. Freedman and Barry F. Madore. 2023. The cepheid extragalactic distance scale: Past, present and future. *arXiv*.

[50] Wendy L. Freedman, Barry F. Madore, Taylor Hoyt, In Sung Jang, Rachael Beaton, Myung Gyoon Lee, Andrew Monson, Jill Neeley, and Jeffrey Rich. 2020. Calibration of the tip of the red giant branch (trgb). *arXiv*.

[51] Markus Freitag and Yaser Al-Onaizan. 2017. Beam search strategies for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 56–60, Vancouver. Association for Computational Linguistics.

[52] Luís P.F. Garcia, André C.P.L.F. de Carvalho, and Ana C. Lorena. 2016. Noise detection in the meta-learning level. *Neurocomputing*, 176:14–25. Recent Advancements in Hybrid Artificial Intelligence Systems and its Application to Real-World Problems.

[53] Aritra Ghosh, Himanshu Kumar, and P. S. Sastry. 2017. Robust loss functions under label noise for deep neural networks. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 1919–1925. AAAI Press.

[54] Lorenzo Giambagli, Duccio Fanelli, Guido Risaliti, and Matilde Signorini. 2023. Nonparametric analysis of the hubble diagram with neural networks. *Astron. Astrophys.*, 678:A13.

[55] Matthew L Ginsberg. 2011. Dr. fill: Crosswords and an implemented solver for singly weighted csps. *Journal of Artificial Intelligence Research*, 42:851–886.

[56] Andrés Felipe Giraldo-Forero, Jorge Alberto Jaramillo-Garzón, and César Germán Castellanos-Domínguez. 2015. Evaluation of example-based measures for multi-label classification performance. In *Bioinformatics and Biomedical Engineering*, pages 557–564, Cham. Springer International Publishing.

[57] Eleonora Giunchiglia and Thomas Lukasiewicz. 2020. Coherent hierarchical multi-label classification networks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

[58] Isidro Gómez-Vargas, Joshua Briones Andrade, and J. Alberto Vázquez. 2023. Neural networks optimized by genetic algorithms in cosmology. *Phys. Rev. D*, 107(4):043509.

[59] Isidro Gómez-Vargas, Ricardo Medel Esquivel, Ricardo García-Salcedo, and J. Alberto Vázquez. 2023. Neural network reconstructions for the hubble parameter, growth rate and distance modulus. *Eur. Phys. J. C*, 83(4):304.

[60] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

[61] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

[62] Yu Han, Donna M. Rizzo, John P. Hanley, Emily L. Coderre, and Patricia A. Prelock. 2022. Identifying neuroanatomical and behavioral features for autism spectrum disorder diagnosis in children using machine learning. *PLOS ONE*, 17(7):1–19.

[63] Awni Y. Hannun, Pranav Rajpurkar, Masoumeh Haghpanahi, Geoffrey H. Tison, Codie Bourn, Mintu P. Turakhia, and Andrew Y. Ng. 2019. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*, 25(1):65–69.

[64] Dhiraj Kumar Hazra, Subhabrata Majumdar, Supratik Pal, Sudhakar Panda, and Anjan A. Sen. 2015. Post-planck dark energy constraints. *Phys. Rev. D*, 91:083005.

[65] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

[66] Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *ArXiv preprint*, abs/1606.08415.

[67] Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. 2018. Using trusted data to train deep networks on labels corrupted by severe noise. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 10477–10486.

[68] John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

[69] John Hewitt, Christopher Manning, and Percy Liang. 2022. Truncation sampling as language model desmoothing. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3414–3427, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

[70] John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

[71] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *ArXiv preprint*, abs/1503.02531.

[72] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

[73] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

[74] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. 2016. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2410–2420, Berlin, Germany. Association for Computational Linguistics.

[75] Wei Huang, Enhong Chen, Qi Liu, Hui Xiong, Zhenya Huang, Shiwei Tong, and Dan Zhang. 2023. Hmcnet: A general approach for hierarchical multi-label classification. *IEEE Trans. Knowl. Data Eng.*, 35(9):8713–8728.

[76] Yi Huang, Buse Giledereli, Arzucan K{oksal, Abdullatif and Özg}ur, and Elif Ozkirimli. 2021. Balancing methods for multi-label text classification with long-tailed class distribution. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8153–8161, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

[77] Seung-gyu Hwang, Benjamin L'Huillier, Ryan E. Keeley, M. James Jee, and Arman Shafieloo. 2023. How to use gp: effects of the mean function and hyperparameter selection on gaussian process regression. *JCAP*, 02:014.

[78] IBM. 20i. Ibm archives: Ibm 1301 disk storage unit. *ArXiv preprint*, abs/i.

[79] Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn L. Ball, Katie S. Shpanskaya, Jayne Seekins, David A. Mong, Safwan S. Halabi, Jesse K. Sandberg, Ricky Jones, David B. Larson, Curtis P. Langlotz, Bhavik N. Patel, Matthew P. Lungren, and Andrew Y. Ng. 2019. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 590–597. AAAI Press.

[80] ISO/IEC 7812-1:2017. 2017. Identification cards — identification of issuers — part 1: Numbering system. Standard, International Organization for Standardization, Geneva, CH.

[81] Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

[82] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna

Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

[83] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *Preprint*, arXiv:2401.04088.

[84] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross B. Girshick. 2017. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1988–1997. IEEE Computer Society.

[85] Ying Ju, Fubang Zhao, Shijie Chen, Bowen Zheng, Xuefeng Yang, and Yunfeng Liu. 2019. Technical report on conversational question answering. *arXiv preprint 1909.10772*.

[86] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. 2021. Highly accurate protein structure prediction with alphafold. *Nature*, 596.

[87] Passent El Kafrawy, Amr Mausad, and Heba Esmail. 2015. Experimental comparison of methods for multi-label classification in different application domains. *International Journal of Computer Applications*, 114(19):1–9.

[88] Davood Karimi, Haoran Dou, Simon K. Warfield, and Ali Gholipour. 2020. Deep learning with noisy labels: Exploring techniques and remedies in medical image analysis. *Medical Image Analysis*, 65:101759.

[89] Divyansh Kaushik, Eduard H. Hovy, and Zachary Chase Lipton. 2020. Learning the difference that makes A difference with counterfactually-augmented data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

[90] Ryan E. Keeley, Arman Shafieloo, Gong-Bo Zhao, Jose Alberto Vazquez, and Hanwool Koo. 2021. Reconstructing the universe: Testing the mutual consis-

tency of the pantheon and sdss/eboss bao data sets with gaussian processes. *Astron. J.*, 161(3):151.

[91] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[92] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023. Segment anything. *ArXiv preprint*, abs/2304.02643.

[93] Terry Koo, Frederick Liu, and Luheng He. 2024. Automata-based constraints for language model decoding. In *First Conference on Language Modeling*.

[94] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114.

[95] Maxat Kulmanov and Robert Hoehndorf. 2022. Deepgozero: improving protein function prediction from sequence and zero-shot learning based on ontology axioms. *Bioinformatics*, 38:i238–i245.

[96] Saurabh Kulshreshtha, Olga Kovaleva, Namrata Shivagunde, and Anna Rumshisky. 2022. Down and across: Introducing crossword-solving as a new NLP benchmark. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2648–2659, Dublin, Ireland. Association for Computational Linguistics.

[97] Ugur Kursuncu, Manas Gaur, and Amit P. Sheth. 2019. Knowledge infused learning (K-IL): towards deep incorporation of knowledge in deep learning. *ArXiv preprint*, abs/1912.00512.

[98] Matt J. Kusner, Joshua R. Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual fairness. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4066–4076.

[99] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. 2020. The open images dataset v4. *International Journal of Computer Vision*, 128(7):1956–1981.

[100] Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. 1989. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems 2, [NIPS Conference, Denver, Colorado, USA, November 27-30, 1989]*, pages 396–404. Morgan Kaufmann.

[101] Tao Li and Vivek Srikumar. 2019. Augmenting neural networks with first-order logic. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 292–302, Florence, Italy. Association for Computational Linguistics.

[102] Xiaolei Li, Ryan E. Keeley, Arman Shafieloo, and Kai Liao. 2024. A model-independent method to determine h $_0$ using time-delay lensing, quasars, and type ia supernovae. *Astrophys. J.*, 960(2):103.

[103] Seppo Linnainmaa. 1976. Taylor expansion of the accumulated rounding error.

[104] Michael L. Littman, Greg A. Keim, and Noam Shazeer. 2002. A probabilistic approach to solving crossword puzzles. *Artificial Intelligence*, 134(1):23–55.

[105] Liang Liu, Li-Juan Hu, Li Tang, and Ying Wu. 2023. Constraining the spatial curvature of the local universe with deep learning. *Res. Astron. Astrophys.*, 23(12):125012.

[106] Yang Liu, Qince Li, Kuanquan Wang, Jun Liu, Runnan He, Yongfeng Yuan, and Henggui Zhang. 2021. Automatic multi-label ecg classification with category imbalance and cost-sensitive thresholding. *Biosensors*, 11(11):453.

[107] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv preprint*, abs/1907.11692.

[108] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. *ArXiv preprint*, abs/2103.14030.

[109] Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. 2017. The expressive power of neural networks: A view from the width. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6231–6239.

[110] Chris Madge and Massimo Poesio. 2024. Large language models as minecraft agents. *ArXiv preprint*, abs/2402.08392.

[111] Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, and Sašo Džeroski. 2012. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9):3084–3104. Best Papers of Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA'2011).

[112] Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, and Marco Gori. 2020. Lyrics: A general interface layer to integrate logic inference and deep learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 283–298, Cham. Springer International Publishing.

[113] Athanasios Masouris and Jan van Gemert. 2023. End-to-end chess recognition. *Preprint*, arXiv:2310.04086.

[114] Ahmad Mehrabi. 2023. A semi-model-independent approach to describe a cosmological database. *arXiv*.

[115] Stefano Melacci, Gabriele Ciravegna, Angelo Sotgiu, Ambra Demontis, Battista Biggio, Marco Gori, and Fabio Roli. 2022. Domain knowledge alleviates adversarial attacks in multi-label classifiers. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 44(12):9944–9959.

[116] Meta. 2024. Introducing meta llama 3: The most capable openly available llm to date.

[117] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *Preprint*, arXiv:2402.06196.

[118] Amina Mollaysa, Alexandros Kalousis, Eric Bruno, and Maurits Diephuis. 2019. Learning to augment with feature side-information. In *Proceedings of The Eleventh Asian Conference on Machine Learning*, volume 101 of *Proceedings of Machine Learning Research*, pages 173–187. PMLR.

[119] Purba Mukherjee, Konstantinos F. Dialektopoulos, Jackson Levi Said, and Jurgen Mifsud. 2024. Late-time transition of m_b inferred via neural networks. *arXiv*.

[120] Purba Mukherjee and Ankan Mukherjee. 2021. Assessment of the cosmic distance duality relation using gaussian process. *Mon. Not. Roy. Astron. Soc.*, 504(3):3938–3946.

[121] Nikhil Muralidhar, Mohammad Raihanul Islam, Manish Marwah, Anuj Karpatne, and Naren Ramakrishnan. 2018. Incorporating prior domain knowledge into deep neural networks. In *IEEE International Conference on Big Data, Big Data 2018, Seattle, WA, USA, December 10-13, 2018*, pages 36–45. IEEE.

[122] B. Novosyadlyj, O. Sergijenko, R. Durrer, and V. Pelykh. 2014. Constraining the dynamical dark energy parameters: Planck-2013 vs wmap9. *JCAP*, 05:030.

[123] Eoin Ó Colgáin and M. M. Sheikh-Jabbari. 2021. Elucidating cosmological model dependence with h_0. *Eur. Phys. J. C*, 81(10):892.

[124] Juan de Dios Rojas Olvera, Isidro Gómez-Vargas, and Jose Alberto Vázquez. 2022. Observational cosmology with artificial neural networks. *Universe*, 8(2):120.

[125] OpenAI. 2021. Chatgpt technical report.

[126] OpenAI. 2023. Gpt-4 techincal report.

[127] Lichess Org. 2013. Lichess open database.

[128] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

[129] G. Paolacci, J. Chandler, and P. G. Ipeirotis. 2010. Running experiments on amazon mechanical turk. *Judgment and Decision Making*, 5(5):411–419.

[130] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

[131] Rafael B. Pereira, Alexandre Plastino, Bianca Zadrozny, and Luiz H.C. Merschmann. 2018. Correlation analysis of performance measures for multi-label classification. *Information Processing And Management*, 54(3):359–369.

[132] John P. Pestian, Chris Brew, Pawel Matykiewicz, DJ Hovermale, Neil Johnson, K. Bretonnel Cohen, and Wlodzislaw Duch. 2007. A shared task involving multi-label classification of clinical free text. In *Biological, translational, and clinical language processing*, pages 97–104, Prague, Czech Republic. Association for Computational Linguistics.

[133] Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622, Online. Association for Computational Linguistics.

[134] Andrei Popescu-Belis, Àlex Atrio, Valentin Minder, Aris Xanthos, Gabriel Luthier, Simon Mattei, and Antonio Rodriguez. 2022. Constrained language models for interactive poem generation. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 3519–3529, Marseille, France. European Language Resources Association.

[135] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

[136] Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

[137] Ryan Riegel, Alexander Gray, Francois Luus, Naweed Khan, Ndivhuwo Makondo, Ismail Yunus Akhalwaya, Haifeng Qian, Ronald Fagin, Francisco Barahona, Udit Sharma, Shajith Ikbal, Hima Karanam, Sumit Neelam, Ankita Likhyani, and Santosh Srivastava. 2020. Logical neural networks. *Preprint*, arXiv:2006.13155.

[138] Adam G. Riess and Louise Breuval. 2023. The local value of h_0. *arXiv*.

[139] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

[140] Frank Rosenblatt. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. *Phychological Review*, 65(6):386–408.

[141] Josh Rozner, Christopher Potts, and Kyle Mahowald. 2021. Decrypting cryptic crosswords: Semantically complex wordplay puzzles as a target for nlp. In *Advances in Neural Information Processing Systems*, volume 34, pages 11409–11421. Curran Associates, Inc.

[142] Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. *ArXiv preprint*, abs/1609.04747.

[143] Andreas Ruepp, Alfred Zollner, Dieter Maier, Kaj Albermann, Jean Hani, Martin Mokrejs, Igor V. Tetko, Ulrich G{uldener and Gertrud Mannhaupt and Martin Münsterk}otter, and Hans-Werner Mewes. 2004. The funcat, a functional annotation scheme for systematic classification of proteins from whole genomes. *Nucleic acids research*, 32 18:5539–45.

[144] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning internal representations by error propagation.

[145] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323.

[146] Abdelrahman Boda Sadallah, Daria Kotova, and Ekaterina Kochmar. 2024. Are llms good cryptic crossword solvers? *Preprint*, arXiv:2403.12094.

[147] Soumadeep Saha, Sutanoya Chakraborty, Saptarshi Saha, and Utpal Garain. 2025. Language models are crossword solvers. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2074–2090, Albuquerque, New Mexico. Association for Computational Linguistics.

[148] Soumadeep Saha, Utpal Garain, Arijit Ukil, Arpan Pal, and Sundeep Khandelwal. 2023. Medtric : A clinically applicable metric for evaluation of multi-label computational diagnostic systems. *PLOS ONE*, 18(8):1–19.

[149] Soumadeep Saha, Utpal Garain, Arijit Ukil, Arpan Pal, and Sundeep Khandelwal. 2024. *DOST–Domain Obedient Self-supervision for Trustworthy Multi Label Classification with Noisy Labels*, pages 117–127. Springer Nature Switzerland, Cham.

[150] Soumadeep Saha, Saptarshi Saha, and Utpal Garain. 2024. VALUED - vision and logical understanding evaluation dataset. *Journal of Data-centric Machine Learning Research*.

[151] Takaya Saito and Marc Rehmsmeier. 2015. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*, 10(3):1–21.

[152] Terence Sanger and Pallavi N. Baljekar. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65 6:386–408.

[153] Robert E. Schapire and Yoram Singer. 2000. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39:135–168.

[154] Hendrik Schuff, Heike Adel, and Ngoc Thang Vu. 2020. F1 is Not Enough! Models and Evaluation Towards User-Centered Explainable Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7076–7095, Online. Association for Computational Linguistics.

[155] Bernhard Schölkopf. 2019. Causality for machine learning. *ArXiv preprint*, abs/1911.10500.

[156] D. M. Scolnic, D. O. Jones, A. Rest, Y. C. Pan, R. Chornock, R. J. Foley, M. E. Huber, R. Kessler, G. Narayan, A. G. Riess, S. Rodney, E. Berger, D. J. Brout, P. J. Challis, M. Drout, D. Finkbeiner, R. Lunnan, R. P. Kirshner, N. E. Sanders, E. Schlafly, S. Smartt, C. W. Stubbs, J. Tonry, W. M. Wood-Vasey, M. Foley, J. Hand, E. Johnson, W. S. Burgett, K. C. Chambers, P. W. Draper, K. W. Hodapp, N. Kaiser, R. P. Kudritzki, E. A. Magnier, N. Metcalfe, F. Bresolin, E. Gall, R. Kotak, M. McCrum, and K. W. Smith. 2020. The Complete Light-curve Sample of Spectroscopically Confirmed SNe Ia from Pan-STARRS1 and Cosmological Constraints from the Combined Pantheon Sample. *ArXiv preprint*, abs/2018a.

[157] Dan Scolnic, Dillon Brout, Anthony Carr, Adam G. Riess, Tamara M. Davis, Arianna Dwomoh, David O. Jones, Noor Ali, Pranav Charvu, Rebecca Chen, Erik R. Peterson, Brodie Popovic, Benjamin M. Rose, Charlotte M. Wood, Peter J. Brown, Ken Chambers, David A. Coulter, Kyle G. Dettman, Georgios Dimitriadis, Alexei V. Filippenko, Ryan J. Foley, Saurabh W. Jha, Charles D. Kilpatrick, Robert P. Kirshner, Yen-Chen Pan, Armin Rest, Cesar Rojas-Bravo, Matthew R. Siebert, Benjamin E. Stahl, and WeiKang Zheng. 2022. The pantheon+ analysis: The full data set and light-curve release. *The Astrophysical Journal*, 938(2):113.

[158] T. Segaran and J. Hammerbacher. 2009. *Beautiful Data: The Stories Behind Elegant Data Solutions*. Theory in practice. O'Reilly Media.

[159] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

[160] Luciano Serafini and Artur S. d'Avila Garcez. 2016. Logic tensor networks: Deep learning and logical reasoning from data and knowledge. In *Proceedings of the 11th International Workshop on Neural-Symbolic Learning and Reasoning (NeSy'16)*, volume 1768 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[161] Rahul Shah, Purba Mukherjee, Soumadeep Saha, Utpal Garain, and Supratik Pal. 2024. Deep learning based recalibration of sdss and desi bao alleviates hubble and clustering tensions. *ArXiv preprint*, abs/2412.14750.

[162] Rahul Shah, Soumadeep Saha, Purba Mukherjee, Utpal Garain, and Supratik Pal. 2024. Ladder: Revisiting the cosmic distance ladder with deep learning approaches and exploring its applications. *The Astrophysical Journal Supplement Series*, 273(2):27.

[163] Torgyn Shaikhina and Natalia A. Khovanova. 2017. Handling limited datasets with neural networks in medical applications: A small-data approach. *Artificial Intelligence in Medicine*, 75:51–63.

[164] Noam M. Shazeer, Michael L. Littman, and Greg A. Keim. 1999. Solving crossword puzzles as probabilistic constraint satisfaction. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence*, AAAI '99/IAAI '99, page 156–162, USA. American Association for Artificial Intelligence.

[165] Ryan Sheatsley, Blaine Hoak, Eric Pauley, Yohan Beugin, Michael J. Weisman, and Patrick McDaniel. 2021. On the robustness of domain constraints. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 495–515, New York, NY, USA. Association for Computing Machinery.

[166] Mattia Silvestri, Michele Lombardi, and Michela Milano. 2021. Injecting domain knowledge in neural networks: A controlled experiment on a constrained problem. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 266–282, Cham. Springer International Publishing.

[167] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[168] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. 2022. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–19.

[169] Liwei Song, Xinwei Yu, Hsuan-Tung Peng, and Karthik Narasimhan. 2021. Universal adversarial attacks with natural triggers for text classification. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3724–3733, Online. Association for Computational Linguistics.

[170] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, et al. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

[171] Rick Stevens, Valerie Taylor, Jeff Nichols, Arthur Barney Maccabe, Katherine Yelick, and David Brown. 2020. AI for science: Report on the department of energy (DOE) town halls on artificial intelligence (AI) for science.

[172] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomput.*, 568(C).

[173] Lichao Sun, Kazuma Hashimoto, Wenpeng Yin, Akari Asai, Jia Li, Philip S. Yu, and Caiming Xiong. 20 2. Adv-bert: BERT is not robust on misspellings! generating nature adversarial samples on BERT. *ArXiv preprint*, abs/ 2003.

[174] Lijuan Sun, Songhe Feng, Tao Wang, Congyan Lang, and Yi Jin. 2019. Partial multi-label learning by low-rank and sparse decomposition. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 5016–5023. AAAI Press.

[175] Mirac Suzgun, Luke Melas-Kyriazi, Suproteem K. Sarkar, Scott Duke Kominers, and Stuart M. Shieber. 2022. The harvard uspto patent dataset: A large-scale, well-structured, and multi-purpose corpus of patent applications. *ArXiv preprint*, abs/2207.04043.

[176] Li Tang, Xin Li, Hai-Nan Lin, and Liang Liu. 2021. Model-independently calibrating the luminosity correlations of gamma-ray bursts using deep learning. *Astrophys. J.*, 907(2):121.

[177] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, and 1369 Others. 2023. Gemini: A family of highly capable multimodal models. *ArXiv preprint*, abs/2312.11805.

[178] Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

[179] Nguyen Thai-Nghe, Zeno Gantner, and Lars Schmidt-Thieme. 2010. Cost-sensitive learning methods for imbalanced data. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

[180] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony

Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

[181] Geoffrey G. Towell and Jude W. Shavlik. 1994. Knowledge-based artificial neural networks. *Artificial Intelligence*, 70(1):119–165.

[182] Arash Vahdat. 2017. Toward robustness against label noise in training deep discriminative neural networks. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5596–5605.

[183] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

[184] Matt Visser. 2005. Cosmography: Cosmology without the einstein equations. *Gen. Rel. Grav.*, 37:1541–1548.

[185] Khuong Vo, Tri Nguyen, Dang Pham, Mao Nguyen, Minh Truong, Trung Mai, and Tho T. Quan. 2010. Combination of domain knowledge and deep learning for sentiment analysis of short and informal messages on social media. *ArXiv preprint*, abs/10.1504.

[186] Phong D. Vo, Alexandru-Lucian Gînsca, Hervé Le Borgne, and Adrian Daniel Popescu. 2015. Effective training of convolutional networks using noisy web images. *2015 13th International Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 1–6.

[187] Eric Wallace, Nicholas Tomlin, Albert Xu, Kevin Yang, Eshaan Pathak, Matthew Ginsberg, and Dan Klein. 2022. Automated crossword solving. In *ACL:2022:long*, pages 3073–3085, Dublin, Ireland. acl.

[188] Ge Wang, Jong Chu Ye, Klaus Mueller, and Jeffrey A Fessler. 2018. Image reconstruction is a new frontier of machine learning. *IEEE Trans Med Imaging*, 37(6):1289–1296.

[189] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2024. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*.

[190] Guo-Jian Wang, Si-Yao Li, and Jun-Qing Xia. 2020. Ecopann: A framework for estimating cosmological parameters using artificial neural networks. *Astrophys. J. Suppl.*, 249(2):25.

[191] Guo-Jian Wang, Xiao-Jiao Ma, Si-Yao Li, and Jun-Qing Xia. 2020. Reconstructing functions and estimating parameters with artificial neural networks: A test with a hubble parameter and sne ia. *Astrophys. J. Suppl.*, 246(1):13.

[192] Huazhen Wang, Xin Liu, Bing Lv, Fan Yang, and Yanzhu Hong. 2014. Reliable multi-label learning via conformal predictor and random forest for syndrome differentiation of chronic fatigue in traditional chinese medicine. *PLOS ONE*, 9(6):1–14.

[193] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

[194] Ya Wang, Dongliang He, Fu Li, Xiang Long, Zhichao Zhou, Jinwen Ma, and Shilei Wen. 2020. Multi-label classification with label graph superimposing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):12265–12272.

[195] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

[196] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

[197] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

[198] Ronald J. Williams and David Zipser. 1995. Gradient-based learning algorithms for recurrent networks and their computational complexity.

[199] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv preprint*, abs/1910.03771.

[200] Georg Wölflein and Ognjen Arandjelović. 2021. Determining chess game state from an image. *Journal of Imaging*, 7(6):94.

[201] Hanbei Xie, Xiaodong Nong, Bin Zhang, Huifeng Wang, Zihao Li, and Nan Liang. 2023. Constraints on cosmological models with gamma-ray bursts in cosmology-independent way. *arXiv*.

[202] Ming-Kun Xie and Sheng-Jun Huang. 2018. Partial multi-label learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 4302–4309. AAAI Press.

[203] Ming-Kun Xie and Sheng-Jun Huang. 2022. Partial multi-label learning with noisy label identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3676–3687.

[204] Xiaozheng Xie, Jianwei Niu, Xuefeng Liu, Zhengsu Chen, Shaojie Tang, and Shui Yu. 2021. A survey on incorporating domain knowledge into deep learning for medical image analysis. *Medical Image Analysis*, 69:101985.

[205] Yaqi Xie, Ziwei Xu, Kuldeep S. Meel, Mohan S. Kankanhalli, and Harold Soh. 2019. Embedding symbolic knowledge into deep networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4235–4245.

[206] Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van den Broeck. 2018. A semantic loss function for deep learning with symbolic knowledge. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5498–5507. PMLR.

[207] Shweta Yadav, Usha Lokala, Raminta Daniulaityte, Krishnaprasad Thirunarayan, Francois Lamy, and Amit Sheth. 2021. "when they say weed causes depression, but it's your fav antidepressant": Knowledge-aware attention framework for relationship extraction. *PLOS ONE*, 16(3):1–18.

[208] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

[209] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

[210] Zijiang Yang, Reda Al-Bahrani, Andrew C. E. Reid, Stefanos Papanikolaou, Surya R. Kalidindi, Wei-keng Liao, Alok Choudhary, and Ankit Agrawal.

2019. Deep learning based domain knowledge integration for small datasets: Illustrative applications in materials informatics. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. Material Science.

[211] Yazhou Yao, Fumin Shen, Guosen Xie, Li Liu, Fan Zhu, Jian Zhang, and Heng Tao Shen. 2020. Exploiting web images for multi-output classification: From category to subcategories. *IEEE Transactions on Neural Networks and Learning Systems*, 31(7):2348–2360.

[212] Bin Zhang, Xiaoyao Xie, Xiaodong Nong, Guangzhen Wang, Zhiguo Xiong, Puxun Wu, and Nan Liang. 2023. Model-independent gamma-ray bursts constraints on cosmological models using machine learning. *arXiv*.

[213] Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. 2019. RAVEN: A dataset for relational and analogical visual reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 5317–5327. Computer Vision Foundation / IEEE.

[214] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. Understanding deep learning (still) requires rethinking generalization. *Commun. ACM*, 64(3):107–115.

[215] Jian-Chen Zhang, Yu Hu, Kang Jiao, Hong-Feng Wang, Yuan-Bo Xie, Bo Yu, Li-Li Zhao, and Tong-Jie Zhang. 2024. A nonparametric reconstruction of the hubble parameter h(z) based on radial basis function neural networks. *Astrophys. J. Suppl.*, 270(2):23.

[216] Wei Emma Zhang, Quan Z Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(3):1–41.

[217] Yinyuan Zhang, Ricardo Henao, Zhe Gan, Yitong Li, and Lawrence Carin. 2018. Multi-label learning from medical plain text with convolutional residual models. In *Proceedings of the 3rd Machine Learning for Healthcare Conference*, volume 85 of *Proceedings of Machine Learning Research*, pages 280–294. PMLR.

[218] Wenting Zhao and Carla P. Gomes. 2021. Evaluating multi-label classifiers with noisy labels. *ArXiv preprint*, abs/2102.08427.

[219] Liang Zhou, Xiaoyuan Zheng, Di Yang, Ying Wang, Xuesong Bai, and Xinhua Ye. 2021. Application of multi-label classification models for the diagnosis of diabetic complications. *BMC Medical Informatics and Decision Making*, 21(1):182.

[220] Ke Zhu and Jianxin Wu. 2021. Residual attention: A simple but effective method for multi-label recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 184–193.

[221] Shuang-Li Zhu, Jie Dong, Chenjing Zhang, Yao-Bo Huang, and Wensheng Pan. 2021. Application of machine learning in the diagnosis of gastric cancer based on noninvasive characteristics. *PLOS ONE*, 15(12):1–13.

[222] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2019. A comprehensive survey on transfer learning. *ArXiv preprint*, abs/1911.02685.

Appendix A
COSMOLOGICAL CONSTRAINTS

[1]Constraint-aware DL techniques enable the application of DL towards solving scientific problems. In this Appendix, we explore such a problem arising from the field of cosmology. In addition to constraints imposed by established laws of physics, this problem presents extra hurdles, such as the samples not being independently and identically distributed (i.i.d.) and the datasets being extremely limited in size. We discuss the process involved in developing a first-of-its-kind technique to tackle these challenges from the ground up and present empirical results to validate our technique.

One of the core challenges in cosmology has been measuring distances. Several methods to indirectly estimate distances to far-flung objects have been devised, each having its own region of operation. In addition, the aptly named *cosmic distance ladder* principle allows us to calibrate the various measurements using the potential regions of overlap between these disparate measurement techniques.

In the conventional distance ladder approach [138], we start with geometric measurements of nearby objects as the first "rung" of the ladder. These measurements are then used to calibrate Cepheid variables [49] or Tip of the Red Giant Branch (TRGB) stars [50]. Finally, using these measurements, we estimate distances to the Type Ia supernovae (SNIa). On the other hand, the "inverse" distance ladder begins with the

[1]This appendix is largely based on our paper titled *"LADDER: Revisiting the Cosmic Distance Ladder with Deep Learning Approaches and Exploring Its Applications"* [162].

Cosmic Microwave Background (CMB), and through various intermediate steps, ultimately extends to SNIa at lower redshifts [16, 24]. SNIa are favored as the final step for both ladders due to their reliability across a wide range of redshifts.[2]

The current standard model of understanding of the evolution of the universe is Λ Cold Dark Matter (ΛCDM), which has six free parameters that must be computed from observational data. However, employing this model alongside observational data has led to some inconsistencies–the most notable of which is the Hubble tension [12, 64, 122]. This, alongside a growing list of other issues has prompted the cosmological community to device more complicated models or look for "model-free" approaches to the cosmic distance ladder paradigm.

Alongside cosmography [184], which utilizes a Taylor series, several other methods like Gaussian processes, genetic algorithms, etc., have been studied in this context [8, 90, 102, 120]. However, ambiguity over polynomial degree, choice of kernels, convergence issues, overfitting concerns, and overwhelming errors in data-scarce regions, have hindered widespread adoption of these techniques [77, 123]. Owing to these shortcoming, DL-based approaches have gained prominence in this domain [30, 31, 40, 54, 58, 59, 105, 114, 119, 124, 176, 190, 191, 201, 212, 215]. Despite several advances, DL-based methods have not been able to extrapolate observational data to data-scarce regions in a consistent manner while taking into account all the available aleatoric noise information. To this end, we device a novel algorithm from first-principles called **LADDER** (**L**earning **A**lgorithm for **D**eep **D**istance **E**stimation and **R**econstruction) which is the first constraint aware DL-based solution to this problem.

In the following sections we first formally state the problem and the associated constraints, followed by a presentation of our solution and elaborate on the design decisions. Finally, we present experimental results validating our approach.

A.1 BACKGROUND

[2] *Redshift* here refers to the frequency of the observed light being reduced owing to the object moving away from the observer as a consequence of the Doppler effect. Edwin Hubble first noted that there is a correlation between the distance to a stellar object and the speed at which it is moving away from the Earth. Given this well-established relationship, distances are often talked about in terms of the amount of redshifts.

DATASETS

Our study features the *Pantheon* [156] and *Pantheon+* [157] SNIa datasets. The *Pantheon* compilation features data from 1048 spectroscopically confirmed SNIa covering a redshift range of $0.01 \lesssim z < 2.3$, with most of the samples being on the lower end of that redshift range. The *Pantheon+* dataset is similar and features data from 1550 distinct SNIa over a redshift range of $0.001 \lesssim z < 2.3$ and has a higher density at the lower redshift range (see Figure A.1). There is some overlap between the datasets, and we *reserve the 753 data points in Pantheon+ that are not in the Pantheon compilation for testing.*

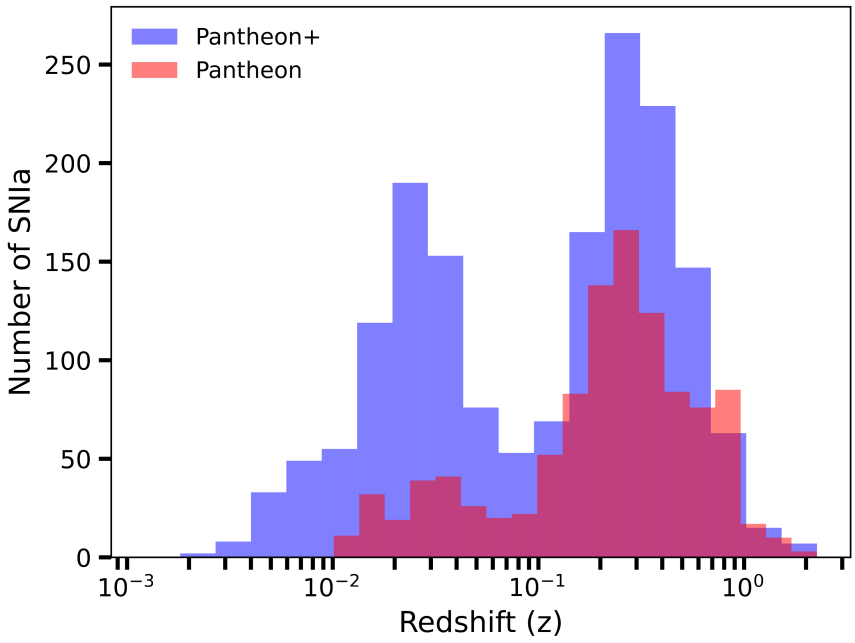


FIGURE A.1: **Distribution of the *Pantheon* [156] and *Pantheon+* [157] datasets.** *Pantheon+* covers a wider range of redshifts with a higher density at lower redshifts. Figure is reproduced from Shah et al. [162].

In addition to the apparent magnitude (m) and the associated error (Δm), at various z, the datasets contain $[\mathbf{C}_{\text{sys}}]_{ij}$ which corresponds to the covariance between sample i and j $\forall i, j$.

FORMAL PROBLEM DESCRIPTION

Given the *Pantheon* dataset, $\mathcal{D} = \{(z_i, m_i, \Delta m_i) | \forall i \in \{1, \dots N\}\}$, where z_i, m_i, and $\Delta m_i \in \mathbb{R}$, which is drawn from some *a priori* unknown distribution, and $\mathbf{C}_{\text{sys}}$, we are interested in estimating the distribution of $\mathcal{P}(M = m|z)$ $\forall$ $z \in \mathbb{R}^+$ with the

assumption $\mathcal{P}(M = m|z) = \mathcal{N}(\mu_\theta(z), \sigma_\theta(z))$, for some functions $\mu_\theta, \sigma_\theta$ and some parameter θ. In the typical empirical risk minimization paradigm, this can be restated as - given $(\mathcal{D}, \mathbf{C}_{\text{sys}})$ find $f : \mathcal{Z} \to \mathbb{R}^2$, such that for any new input z, we have:

$$\min_{f \in \mathcal{F}} \mathcal{E}[f] \,, \tag{A.1}$$

$$\mathcal{E}[f] = \frac{1}{N} \sum_{i=1}^{N} \ell\Big(\mathcal{N}(m_i, \Delta m_i), \mathcal{N}([f(z_i)]_1, [f(z_i)]_2)\Big) \tag{A.2}$$

for some class of functions $\mathcal{F}$ and loss ℓ. $\mathcal{E}$ is the empirical risk functional. We choose ℓ to be the KL-divergence since we seek to model the distribution of $m(z)$.

Although we seek to interpolate from the dataset, this is not a standard regression problem since we have:

$$\mathcal{P}(m_1, m_2, \ldots, m_N | z_1, z_2, \ldots, z_N) \neq \prod_{i=1}^{N} \mathcal{P}(m_i | z_i) \tag{A.3}$$

Thus, we have to contend with the following intractable empirical risk:

$$\begin{aligned} \mathcal{E}^{\text{emp}}[f] = &\ell\Big(\mathcal{N}\big((m_1, m_2, \ldots m_N), \Sigma_m\big), \\ &\mathcal{N}\big(([f(z_1)]_1, [f(z_2)]_1, \ldots, [f(z_N)]_1), \Sigma_f\big)\Big) \end{aligned} \tag{A.4}$$

for some covariance matrix Σ_m, Σ_f. We also have a few constraints based on known laws of physics–(i) $m(z)$ is monotonic, i.e., $m(z_1) \geq m(z_2) \ \forall z_1 \geq z_2$, and (ii) $m(z)$ is at least once differentiable.

A.2 LADDER ALGORITHM

We first make a simplifying assumption to deal with the empirical loss given in Equation A.4. We assume that at most K samples from the dataset are correlated and rewrite the empirical risk as:

$$\begin{aligned} \mathcal{E}^{\text{emp}} \approx \ \frac{1}{\binom{N}{K}} \sum_{\substack{\text{all combinations} \\ j_1, \ldots j_K}} &\ell\Big(\mathcal{N}\big((m_{j_1}, \ldots m_{j_K}), \Sigma_m^K\big), \\ &\mathcal{N}\big(([f(z_{j_1})]_1, \ldots [f(z_{j_K})]_1), \Sigma_f^K\big)\Big) \end{aligned} \tag{A.5}$$

where Σ_m^K is the $K \times K$ sub-matrix of covariances, and $[\Sigma_m^K]_{u,v} = [\Sigma_m]_{j_u,j_v}$. Note that this approach does not ignore other correlations, since we minimize risk over every possible combination of samples from the dataset. This motivates our choice of the modeling function to be of the form:

$$\begin{aligned} &f_\theta : \mathbb{R}^{2\cdot K-1} \to \mathbb{R} \times \mathbb{R}^+ \,, \\ &f_\theta(z; z_{j_1}, z_{j_2}, \dots z_{j_{K-1}}, m_{j_1}, m_{j_2}, \dots m_{j_{K-1}}) \mapsto (\mu_z, \sigma_z), \\ &\mathcal{N}(\mu_z, \sigma_z) \approx \mathcal{P}(m|z; z_{j_1}, z_{j_2}, \dots z_{j_{K-1}}, m_{j_1}, m_{j_2}, \dots m_{j_{K-1}}). \end{aligned} \tag{A.6}$$

Our objective then is to minimize:

$$\begin{aligned} \mathcal{E}^{\text{emp}}[f_\theta] \approx \frac{1}{\binom{N}{K}} \sum_{\substack{\text{all combinations} \\ j_1,\dots j_K}} \sum_{k=1}^{N} \ell\Big(\mathcal{N}\big(m_k, \Delta m_k\big), \\ \mathcal{N}\big(f_\theta(z_k; z_{j_1}, z_{j_2}, \dots m_{j_{K-1}})\big)\Big), \end{aligned} \tag{A.7}$$

$$\ell(P, Q) = D_{KL}(P||Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)} \qquad \text{(KL divergence)}$$

This simplified problem can now be addressed with standard DL techniques.

Algorithm 3 : LADDER

Given $\mathcal{D}$, $\mathbf{C}_{\text{sys}}$, and batch size B.
Initialize θ_0.
while not StopCondition **do**
  $l \leftarrow 0$
  for $i = 1, 2, \dots B$ **do**
    Get K samples from $\mathcal{D} \to \{(z_1, m_1, \Delta m_1), \dots (z_K, m_K, \Delta m_K)\}$
    Sample $\hat{m}_{j_2}, \hat{m}_{j_3}, \dots \hat{m}_{j_K} \sim \mathcal{N}(m_{j_2}, m_{j_3}, \dots m_{j_K}, \Sigma_m^K)$ ▷ see Equation A.8
    $X_i = ((z_{j_2}, \hat{m}_{j_2}), (z_{j_3}, \hat{m}_{j_3}), \dots, (z_{j_K}, \hat{m}_{j_K}); z_{j_1})$
    $Y_i = (m_{j_1}, \Delta m_{j_1})$
    $\mu \leftarrow [f_{\theta_t}(X_i)]_1, \sigma \leftarrow [f_{\theta_t}(X_i)]_2$ ▷ Forward pass.
    $l \mathrel{+}= D_{KL}\Big(\mathcal{N}(m_{j_1}, \Delta m_{j_1}), \mathcal{N}(\mu, \sigma)\Big)$
  end for
  Compute $\nabla_{\theta_t} l \quad \forall \theta_t$
  $\theta_{t+1} \leftarrow \text{Adam}(\nabla_{\theta_t}, \theta_t, \epsilon, \beta_1, \dots)$ ▷ Gradient update.
  if ... **then** ▷ Check for convergence.
    StopCondition $\leftarrow$ True
  end if
end while

The training algorithm is outlined in Algorithm 3. During training, we choose K samples from $\mathcal{D}$ and randomly designate $K-1$ of them as *"support"* and the remaining sample as *query*. To take the covariances into account, instead of directly using these data points, we sample from the joint distribution $\mathcal{N}(m_{j_2}, m_{j_3}, \dots m_{j_K}, \Sigma_m^K)$ to obtain $\hat{m}_{j_2}, \hat{m}_{j_3}, \dots \hat{m}_{j_K}$, where Σ_m^K is given by:

$$
\begin{aligned}
[\Sigma_m]_{\alpha,\beta} &= \begin{cases} [\mathbf{C}_{\text{sys}}]_{\alpha,\beta} \text{ if } \alpha \neq \beta \\ (\Delta m_\alpha)^2 \text{ if } \alpha = \beta \end{cases} \\
[\Sigma_m^K]_{u,v} &= [\Sigma_m]_{j_u, j_v} \quad \forall u, v \in \{1, \dots, K-1\}
\end{aligned}
\tag{A.8}
$$

Following this, we reorder the samples to be in sorted order, i.e., $z_{j_a} \leq z_{j_b}$ whenever $j_a < j_b$, as this was found to aid adherence to the *monotonicity constraint*. [3] Finally, we construct a training instance $X = ((z_{j_2}, \hat{m}_{j_2}), (z_{j_3}, \hat{m}_{j_3}), \dots, (z_{j_K}, \hat{m}_{j_K}); z_{j_1})$ and $Y = (m_{j_1}, \Delta m_{j_1})$ and use the Adam optimizer [91] to train a NN with the following objective:

$$
\ell_{\text{instance}}(X, Y) = D_{KL}\Big(\mathcal{N}(m_{j_1}, \Delta m_{j_1}) \,\Big|\Big|\, \mathcal{N}([f_\theta(X)]_1, [f_\theta(X)]_2)\Big) \tag{A.9}
$$

$$
\hat{\theta} = \arg\min_\theta \mathbb{E}[\ell_{\text{instance}}] \tag{A.10}
$$

During inference, given z, we sample S sets of $K-1$ points from $\mathcal{D}$ and consider the joint distributions $\mathcal{N}((m_{j_2}^{(i)}, \dots, m_{j_{K-1}}^{(i)}), \Sigma_m^K)\, \forall i \in \{1, \dots, S\}$. From each joint distribution, we sample $(\hat{m}_{j_2}^{(i)}, \dots, \hat{m}_{j_{K-1}}^{(i)})$ to create

$$
X_{\text{unseen}}^{(i)} = ((z_{j_1}^{(i)}, \hat{m}_{j_1}^{(i)}), \dots, (z_{j_{K-1}}^{(i)}, \hat{m}_{j_{K-1}}^{(i)}); z_{\text{unseen}}) \tag{A.11}
$$

We then use the trained NN $f_{\hat{\theta}}$, to compute $\mu^{(i)}, \sigma^{(i)}$. We wish to model $\mathcal{P}(m|z) = \int_{z_1,\dots,z_K} \mathcal{P}(m|z; z_{j_1}, \dots z_{j_{K-1}}, m_{j_1}, \dots m_{j_{K-1}}) d\mu$, which can be approximated by the Monte Carlo method using the definition in Equation A.6 as follows:

$$
\begin{gathered}
\mu \leftarrow \frac{1}{S}\sum_{i=1}^{S}[f_{\hat{\theta}}(X_{\text{unseen}}^{(i)})]_1, \quad \sigma \leftarrow \frac{1}{S}\sum_{i=1}^{S}[f_{\hat{\theta}}(X_{\text{unseen}}^{(i)})]_2 \\
\mathcal{P}(m|z) \approx \mathcal{N}(\mu, \sigma)
\end{gathered}
\tag{A.12}
$$

This proxy objective asks–based on $K-1$ (support) points from the dataset–to predict $(m, \Delta m)$ corresponding to the point of interest (query). This approach is

[3] The function f_θ is always at least once differentiable almost everywhere in DL.

robust to perturbations because we not only sample from the joint normal distribution to generate inputs, but also randomly vary the support points corresponding to each query point. Another consideration is *consistency*, i.e., if we re-instantiate the algorithm with different random seeds, our predictions should not change considerably, and this approach was found to satisfy this criterion.

A.3 EXPERIMENTS

Our network of choice for training with the objective in Equation A.10 is an LSTM [72], and we employed a 2-layer LSTM, with a further FC layer projecting the final hidden state to $\mathbb{R}^2$. At the outset, we perform some validation experiments with a $80\% - 20\%$ held-out split of the *Pantheon* dataset. In addition to performing ablation tests of our algorithm, we check performance in three key metrics of importance:

1. MSE on the held-out validation set.
2. *Monotonicity*–given by Spearman correlation.
3. *Smoothness* defined as $\text{Smoothness}[f_\theta] := \frac{1}{n}\sum_{i=1}^{n} |f''_\theta(z_i)|$.

Additionally, we also tested MLPs and other ML techniques like k-nearest neighbors (kNN) and support vector regression (SVR) to serve as baselines.

TABLE A.1: **Performance of various ML models.**
In addition to MSE on the held-out set, we study constraint adherence *vis-à-vis* smoothness and monotonicity. [↓] indicates lower is better, [↑] indicates higher is better.

Model	**MSE** [↓]	**Monotonicity** [↑]	**Smoothness** [↓]
kNN (k=5)	0.022116	0.99999	90.67500
SVR	0.019358	**1.0**	3.10633
MLP (K=1)	0.022202	**1.0**	**2.21691**
MLP (K=32)	0.020484	0.99997	88.99974
LADDER	**0.018495**	**1.0**	2.30022

We start by examining the effect of the parameter K (see Figure A.2, Table A.1), i.e., *support* size, on the performance of tested models and found that the LSTM with $K = 32$ performs best in terms of error on the validation set. The $K = 1$ MLP, i.e., standard regression without *LADDER*, was the smoothest, followed closely by

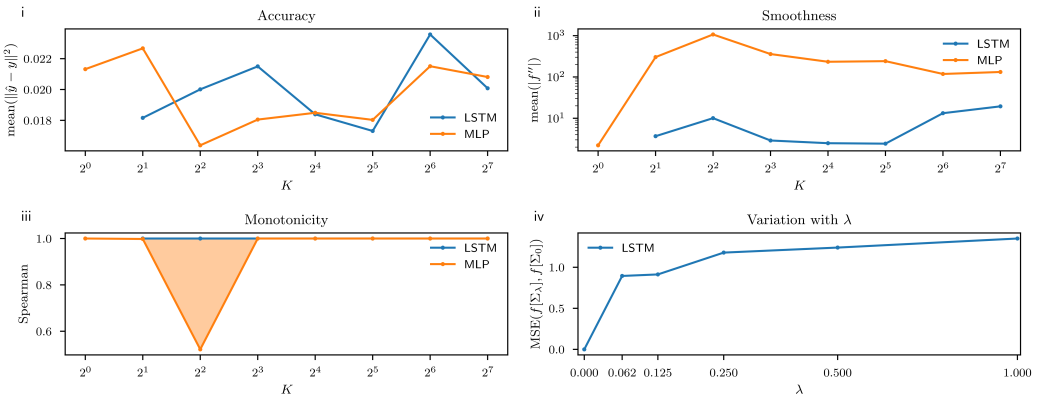


FIGURE A.2: **Results of ablation and validation experiments with various models.**
Panels i, ii, and iii show variation of error, smoothness, and monotonicity performance with a changing value of K, respectively. MLP models do not produce smooth, monotonic results (except K=1), and the $K = 1$ MLP is outperformed by the LSTM model at roughly the same smoothness and monotonicity. *Panel iv* shows the variation in the prediction as measured by MSE between models trained with Σ_λ and Σ_0. When the covariance matrix is progressively corrupted with noise, the predictions change, thus demonstrating our approach's ability to model correlations. Figure is reproduced from Shah et al. [162].

the LSTM trained with *LADDER*. Although monotonicity violations are rare, the MLP and kNN approaches underperform in this regard. We conclude that the LSTM network with $K = 32$ trained with the *LADDER* approach is the best performer overall.

To ablate the effect of the covariance matrix, i.e., to investigate whether *LADDER* can effectively model the correlation information, we construct:

$$\Sigma_\lambda = \lambda \mathbf{N} + (1-\lambda)\mathbf{C}_{\text{sys}} + \mathbb{I}_{N\times N} \cdot \Big((\Delta m_1)^2, (\Delta m_2)^2, \dots (\Delta m_N)^2 \Big) \tag{A.13}$$

where $\mathbf{N} := A \times A^T$ is by construction a symmetric positive semi-definite matrix, such that $A_{ij} \sim \mathcal{N}(0,1)$ is Gaussian noise. Note, $\lambda = 0$ corresponds to the no-noise case and is equivalent to Σ_m as defined in Equation A.8. We then train LSTMs following the *LADDER* algorithm with varying λ, i.e., varying levels of noise added to the non-diagonal elements of the covariance matrix, and measure the effect of noise on the final network predictions. We find (see Figure A.2) that predictions vary consistently with increased amounts of noise, thus suggesting that the *LADDER* algorithm imbues this correlation information into the network.

Finally, we compare the performance of the LSTM network trained with the *LADDER* approach on the unseen *Pantheon+* dataset (see Figure A.3). Our algorithm accurately models the distribution of apparent magnitude data over a wide range of redshifts with consistency and constraint adherence while taking into account sample correlations.

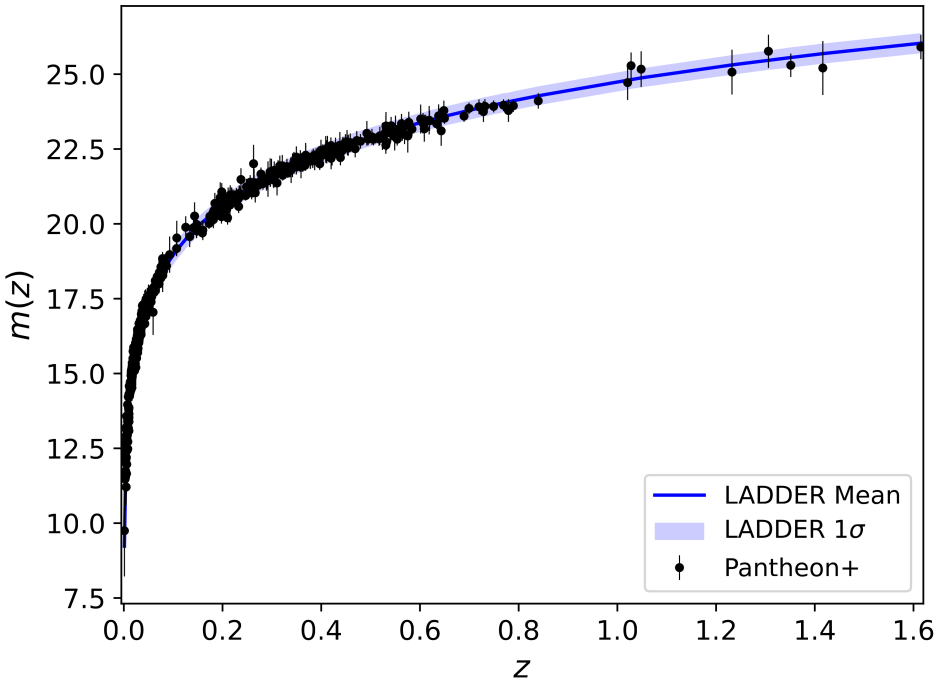


FIGURE A.3: *LADDER* predictions compared to the *Pantheon+* dataset (unseen). Figure is reproduced from Shah et al. [162].

Successfully modeling the distribution of $m(z)$ over a wide range of redshifts has various cosmological implications, such as being useful to calibrate other datasets following the *cosmic distance ladder* paradigm, or serving as a mock data generator [161, 162]. A detailed discussion on further cosmological implications is outside the scope of this dissertation and can be found in Shah et al. [162].