Recursive Bayesian Networks: Generalising and Unifying Probabilistic Context-Free Grammars and Dynamic Bayesian Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Probabilistic context-free grammars (PCFGs) and dynamic Bayesian networks (DBNs) are widely used sequence models with complementary strengths and limitations. While PCFGs allow for nested hierarchical dependencies (tree structures), the latent variables (non-terminal symbols) have to be discrete. In contrast, DBNs allow for continuous latent variables, but the dependencies are strictly sequential (chain structure). Therefore, neither can be applied if the latent variables are assumed to be continuous and *also* have a nested hierarchical dependency structure. In this paper, we present Recursive Bayesian Networks (RBNs), which generalise and unify PCFGs and DBNs, combining their strengths and containing both as special cases. RBNs define a joint distribution over tree-shaped Bayesian networks with continuous latent variables. The main challenge lies in performing *joint* inference over the exponential number of possible structures and the continuous variables. We provide two solutions: 1) For arbitrary RBNs, we generalise inside and outside probabilities from PCFGs to the continuous case, which allows for maximum posterior estimates of the latent variables via gradient descent, while marginalising over network structures. 2) For Gaussian RBNs, we additionally derive an analytic approximation for the marginal likelihood (evidence) and marginal posterior distribution, allowing for robust parameter optimisation and Bayesian inference. The capacity and diverse applications of RBNs are illustrated on two examples: In a quantitative evaluation on synthetic data, we demonstrate and discuss the advantage of RBNs for segmentation and tree induction from noisy sequences, compared to change point detection and hierarchical clustering. In an application to musical data, we approach the unsolved problem of hierarchical music analysis from the raw note level and compare our results to expert annotations.

1 Introduction

Long-term dependencies with a nested hierarchical structure are one of the major challenges in modelling sequential data. This type of dependencies is common in many domains, such as natural language [1], music [2, 3], or decision making [4, 5]. Two of the most widely used probabilistic models for sequential data are probabilistic context-free grammars (PCFGs) and dynamic Bayesian networks (DBNs), both having complementary strengths.

PCFGs are well-established and widely used for modelling hierarchical long-term dependencies in symbolic data [1, 6, 7, 8, 9, 5, 10]. They generalise local (Markov) transition models by allowing for

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

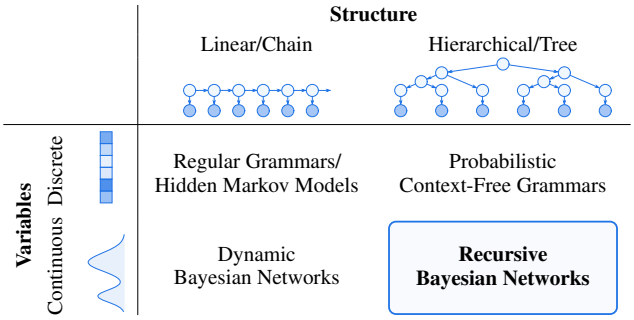

Figure 1: RBNs generalise PCFGs by allowing for continuous latent variables and DBNs by incorporating nested hierarchical dependencies.

infinitely many levels of nested hierarchical dependencies and a flexible number of latent variables. However, parsing methods, such as the Cocke-Younger-Kasami (CYK) algorithm [11, 12, 13, 14, 6] rely on the discrete nature of the rules and variables.

In contrast, DBNs are sequential models with a fixed set of random variables that reoccur at each time step [15, 16]. The variables may be discrete or continuous, latent or observed, and may have an arbitrary non-cyclic dependency structure among each other with additional links from and to the previous and next time slice, respectively. They comprise important model classes as special cases, such as hidden Markov models (HMMs) if there is only a single discrete latent variable or linear dynamical systems if all dependencies are linear Gaussians [17, 18, 16]. However, DBNs only allow for a fixed chain of Markov dependencies between time slices and cannot represent nested hierarchical structures.

In this paper, we present Recursive Bayesian Networks (RBNs), a novel class of probabilistic models that combines the strengths of PCFGs and DBNs by allowing for nested hierarchical dependencies in combination with arbitrary discrete or continuous random variables (Figure 1). Our main contributions are as follows:

1. With RBNs, we provide a unified theoretical framework for a large class of important sequence models, including PCFGs and DBNs.
2. We generalise inside and outside probabilities to continuous latent variables, allowing for maximum posterior (MAP) inference in arbitrary RBNs.
3. For Gaussian RBNs, we derive an analytic approximation for the marginal likelihood and marginal posterior distribution, allowing for robust parameter optimisation and Bayesian inference.
4. We provide a quantitative evaluation on synthetic data and an application to the challenging task of hierarchical music analysis.

1.1 Related Work

Probabilistic context-free grammars (PCFGs) have a long tradition for modelling nested hierarchical dependencies in symbolic data with a variety of parsing algorithms for inferring the dependency structure and the variables' values, both in a deterministic and a probabilistic setting [13, 6, 14]. Beyond their application to sequential data, PCFGs have been generalised to graphs structures [19, 20, 21, 22], which readily transfer to applications of RBNs. More recently, the rise of libraries with automatic differentiation capabilities, such as `PyTorch` [23], has lead to a number of applications where gradients are propagated through the entire parsing process [24, 25, 26, 27]. Our method for realising inference in RBNs with continuous variables is in line with these approaches. Recursive neural tensor networks [28] combine a PCFG with continuous variable by first parsing a sentence to obtain a tree structure and then using it as the backbone for a neural network. However, this is a cleanly separate two-step process (first discrete, then continuous) and the continuous variables do not influence the tree structure.

Structure Inference in Graphical Models: Graphical models are usually assumed to have a fixed structure, possibly conditional on the given data as in template-based models [16]. Some approaches

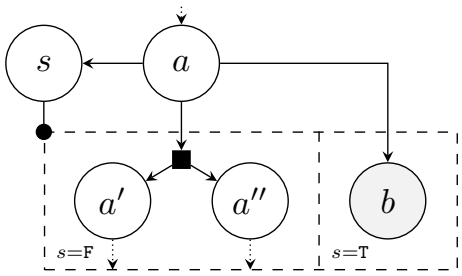

Figure 2: RBN in Chomsky normal form. The non-terminal transition $p_{\mathrm{N}}(a', a'' \mid a)$ and terminal transition $p_{\mathrm{T}}(b \mid a)$ are grouped into an RBN cell with a structural distribution $p_{\mathrm{S}}(s \mid a)$. We use gates [31] to describe structural distributions and extended factor graph notation [black squares; 32] for conditional joint distributions. Considering all possible ways how n observations can be generated by recursively applying the RBN cell produces an RBN chart, as shown in Figure 3.

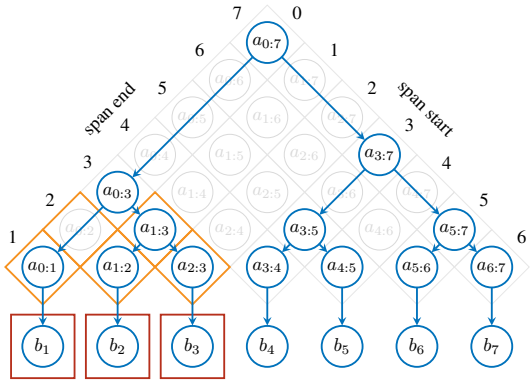

Figure 3: Chart of an RBN in CNF for sequential data of length $n = 7$. The network obtained by fixing one specific dependency structure is highlighted in blue; latent non-terminal variables that are not part of this particular structure are shown in grey. Orange and red boxes, respectively, indicate the subsets $\mathbf{A}_{0:3}$ and $\mathbf{B}_{0:3}$ of latent non-terminal and observed terminal variables generated from $a_{0:3}$.

assume a fixed set of existing variables but attempt to infer the dependency structure among them, which includes the possibility of marginalising over all possible model structures [16, 29, 30]. In the case of RBNs, we are facing a more general problem where additionally the set of variables is unknown and we need to perform inference over both the set of variables and their dependencies. This is realised by drawing the connection to PCFGs and explicitly defining a weighted mixture over Bayesian networks. Therefore, RBNs also provide a novel way to perform structure inference in graphical models.

2 Recursive Bayesian Networks

RBNs are *template-based* graphical models that define a joint distribution over network structures and variables' values. The number of template variables is fixed but the number of instantiated variables, their connectivity, and values are governed by the joint distribution. As a rough analogy, RBNs can be thought of as DBNs that can not only be connected linearly to form a chain but also hierarchically to form a tree structure. Alternatively, they can be thought of as a PCFG in which each symbol also is a continuous random variable.

2.1 Definition

RBNs have three types of template variables: 1) latent non-terminal variables (discrete or continuous), 2) observed terminal variables (discrete or continuous), and 3) latent structural variables (always discrete). In the simplest case, illustrated in Figure 2, an RBN has one template variable of each type. Formally, an RBN is defined as follows:

Definition 1 (Recursive Bayesian Network)**.** *An RBN is a tuple* $(A, B, S, p_{\mathrm{P}}, p_{\mathrm{T}}, p_{\mathrm{S}})$ *with*

$$A: \textit{ set of latent non-terminal template variables} \tag{1}$$
$$B: \textit{ set of observed terminal template variables} \tag{2}$$
$$S: \textit{ set of latent structural template variables, paired up with the non-terminal variables} \tag{3}$$
$$p_{\mathrm{P}}: \textit{ a prior/start distribution for exactly one non-terminal variable} \tag{4}$$
$$p_{\mathrm{T}}: \textit{ a set of transition distributions } p_i(r_i^1, \ldots, r_i^{N_i} \mid l_i) \textit{ a single non-terminal variable } l_i \tag{5}$$
$$p_{\mathrm{S}}: \textit{ a set of structural distributions } p_{\mathrm{S}}(s \mid a)\textit{, one for each non-terminal/structural pair.} \tag{6}$$

The left-hand side l_i *of a transition* p_i *must be a non-terminal variable, while the right-hand side* $r_i^1, \ldots, r_i^{N_i}$ *may consist of any combination of* N_i *non-terminal and terminal variables (*N_i *is the*

arity of the ith transition). The cardinality of the structural variables matches the number of transition distributions from the associated non-terminal variables.

Generating with an RBN is straight forward. We start by sampling the value of the first non-terminal variable a from the prior distribution $p_{\mathrm{P}}(a)$ and then repeat the following steps until no unprocessed non-terminal variables are left:

1. sample the value of the associated structural variable from $p_{\mathrm{S}}(s \mid a)$
2. choose a transition distribution $p_{\mathrm{T}}(r^1, \ldots, r^N \mid a)$ based on the structural variable's value
3. sample all variables $r^1, \ldots, r^N$ from the transition distribution
4. for all newly generated non-terminal variables, go to step 1.

The major challenge and focus of this paper is to perform joint inference over the latent structure and non-terminal variables' values conditional on a given set of observations.

Chomsky Normal Form: In the simplest non-trivial case, an RBN has one latent non-terminal, one latent structural and one observed template variable with one non-terminal transition of arity two and one terminal transition of arity one, as illustrated in Figures 2 and 3. It is defined by four distributions

$$p_{\mathrm{P}}(a) : \text{prior/start distribution} \quad (7) \qquad p_{\mathrm{N}}(a', a'' \mid a) : \text{non-terminal transition} \quad (8)$$

$$p_{\mathrm{T}}(b \mid a) : \text{terminal transition} \quad (9) \qquad p_{\mathrm{S}}(s \mid a) : \text{termination probability .} \quad (10)$$

In analogy to PCFGs, we call this the Chomsky normal form (CNF). Any RBN may be rewritten in CNF (see Appendix for details).

RBN Chart: During inference, we will make use of an RBN *chart*, similar to the parse chart for PCFGs [6]. Each non-terminal variable is associated to one or more layers in the chart. For discrete variables, they store the actual distributions, while for continuous variables they either hold the point estimate (in MAP inference) or the parameters of the approximate distributions (for inference in Gaussian RBN). Different instances of the same template variable are identified by a subscript indicating the span of data generated from them. Sets of variables that may be generated from a specific latent non-terminal variable $a_{i:k}$ are denoted by a bold capital letter with a corresponding subscript ($\mathbf{A}_{i:k}$, $\mathbf{B}_{i:k}$, $\mathbf{S}_{i:k}$). The subscripts are to be interpreted as time intervals, that is, $\mathbf{B}_{i:i}$ is empty, $\mathbf{B}_{0:1} = b_1$ is the first observation, $\mathbf{B}_{n\text{-}2:n} = (b_{n-1}, b_n)$ are the last two observations etc.

Comparison to PCFGs: Any PCFG can be rewritten as a RBN in two different ways, which we call *abstraction* and *expansion*. Abstraction of a PCFG produces a discrete RBN with one latent non-terminal and one observed variable. The resulting RBN is exactly equivalent to the original PCFG but describes the same relations in a more abstract and compact way. In contrast, *expansion* of a PCFG considers the symbols of the grammar as random variables in its own right, thereby endowing them with additional (possibly continuous) degrees of freedom. The resulting RBN is therefore more powerful than the original PCFG. A PCFG is abstracted to a discrete RBN by defining the start, transition, and structural distributions (7–10) as

$$p_{\mathrm{P}}(a{=}A) = \frac{W_{S\to A}}{\sum_{A'} W_{S\to A'}} \quad (11) \qquad p_{\mathrm{N}}(a'{=}B, a''{=}C \mid a{=}A) = \frac{W_{A\to BC}}{\sum_{B',C'} W_{A\to B'C'}} \quad (12)$$

$$p_{\mathrm{T}}(b{=}b \mid a{=}A) = \frac{W_{A\to b}}{\sum_{b'} W_{A\to b'}} \quad (13) \qquad p_{\mathrm{S}}(s \mid a{=}A) = \begin{cases} \frac{\sum_{B,C} W_{A\to BC}}{\sum_X W_{A\to X}} & \text{if } s{=}\mathtt{F} \\ \frac{\sum_b W_{A\to b}}{\sum_X W_{A\to X}} & \text{if } s{=}\mathtt{T}\ , \end{cases} \quad (14)$$

where $A, B, C \in N$ are non-terminal symbols, $b \in T$ is a terminal symbol, $X \in N^2 \cup T$ is any right-hand side of a rule, `T` and `F` are Boolean `true` and `false`, $W_{Y\to Z}$ is the weight of the corresponding rule, and rules that do not exist in the original PCFG are taken to have zero weight (see Appendix for a full proof). In *expansion*, the PCFG is only used to define as a "skeleton" for the RBN, while the specific random variables and the concrete transition distributions need to be additionally specified. This means that the resulting RBN model is more complex than the original PCFG, as the symbols may, for instance, be expanded to continuous random variables (see Appendix for details).

2.2 Inference

The two main goals of inference in RBNs are to 1) train model parameters by maximising the marginal data likelihood and 2) compute posterior distributions or maximum posterior (MAP) estimates of

the network structure and non-terminal variables. In PCFGs, both is achieved by computing inside and outside probabilities [14], which we will be starting point for our generalisation to continuous variables.

Inside and Outside Probabilities: We define inside and outside probabilities, β and α, for RBNs in analogy to how they are defined for PCFGs [14], the only difference being that the variables may be continuous

$$\beta_{i:k}(a_{i:k}) := p(\mathbf{B}_{i:k} \,|\, a_{i:k}) \qquad \text{and} \qquad \alpha_{i:k}(a_{i:k}) := p(\mathbf{B}_{0:i}, a_{i:k}, \mathbf{B}_{k:n}) \,, \tag{15}$$

where n is the length of the sequence. That is, $\beta_{i:k}(a_{i:k})$ is the marginal likelihood of generating the sub-sequence $\mathbf{B}_{i:k}$ conditional on the respective non-terminal variable $\alpha_{i:k}$, while $\alpha_{i:k}(a_{i:k})$ is marginal likelihood of generating the two sub-sequences $\mathbf{B}_{0:i}$ and $\mathbf{B}_{k:n}$ as well as the non-terminal variables $\alpha_{i:k}$. In both cases, β and α are functions of the corresponding non-terminal variable with the remaining variables being marginalised out. Based on the inside and outside probabilities, the marginal data likelihood and the marginal posterior distributions over non-terminal variables are

$$p(\mathbf{B}) = \int \beta_{0:n}(a_{0:n})\, p_{\mathrm{P}}(a_{0:n})\, da_{0:n} \quad (16) \quad \text{and} \quad \widetilde{p}(a_{i:k} \,|\, \mathbf{B}) = \frac{\alpha_{i:k}(a_{i:k})\, \beta_{i:k}(a_{i:k})}{p(\mathbf{B})} \,, \tag{17}$$

respectively, where $\widetilde{p}(a_{i:k} \,|\, \mathbf{B})$ is an unnormalised probability distribution that simultaneously specifies the probability of $a_{i:k}$ to exist and its distribution for the case that it *does* exists.

Inside probabilities are recursively computed bottom-up. For an RBN in CNF we start with the base case (18) for single observations and then iterate (19) to the top of the RBN chart

$$\beta_{i:i+1}(a_{i:i+1}) = p_{\mathrm{S}}(s_{i:i+1}{=}\mathtt{T} \,|\, a_{i:i+1})\, p_{\mathrm{T}}(b_{i+1} \,|\, a_{i:i+1}) \tag{18}$$

$$\beta_{i:k}(a_{i:k}) = p_{\mathrm{S}}(s_{i:k}{=}\mathtt{F} \,|\, a_{i:k}) \sum_{j=i+1}^{k-1} \iint p_{\mathrm{N}}(a_{i:j}, a_{j:k} \,|\, a_{i:k})\, \beta_{i:j}(a_{i:j})\, \beta_{j:k}(a_{j:k})\, da_{i:j}\, da_{j:k} \,. \tag{19}$$

The main conceptual difference to PCFGs is that we treat the (discrete) structural part separately from the (continuous) variables, whereas in PCFGs both are jointly described by the set of rules. For RBNs that are not in CNF, such as our Gaussian RBN for music (Section 2.3.1), (18) and (19) need to be combined by summing over the structural variable s and different possible transitions. The outside probabilities are recursively computed top-down, while making use of the inside probabilities

$$\alpha_{0:n}(a_{0:n}) = p_{\mathrm{P}}(a_{0:n}) \tag{20}$$

$$\alpha_{j:k}(a_{j:k}) = \Big[\sum_{i=0}^{j-1} \iint p_{\mathrm{S}}(s_{i:k}{=}\mathtt{F} \,|\, a_{i:k})\, p_{\mathrm{N}}(a_{i:j}, a_{j:k} \,|\, a_{i:k})\, \alpha_{i:k}(a_{i:k})\, \beta_{i:j}(a_{i:j})\, da_{i:j}\, da_{i:k}\Big] + \Big[\sum_{l=k+1}^{n} \iint p_{\mathrm{S}}(s_{j:l}{=}\mathtt{F} \,|\, a_{j:l})\, p_{\mathrm{N}}(a_{j:k}, a_{k:l} \,|\, a_{j:l})\, \alpha_{j:l}(a_{j:l})\, \beta_{k:l}(a_{k:l})\, da_{j:l}\, da_{k:l}\Big] \,. \tag{21}$$

As for PCFGs, the two terms in (21) correspond to the possibility of $a_{j:k}$ being generated as the right or the left child, respectively. As for the inside probabilities, in the general case, the equations have to be to be adapted accordingly (see Appendix).

Maximum Posterior Inference: For a specific assignment of the non-terminal variables $\mathbf{A}$, the integrals in (18–21) vanish and we can compute the joint marginal likelihood $p(\mathbf{A}, \mathbf{B})$ over observed terminal and latent non-terminal variables by only marginalising over the network structure. $p(\mathbf{A}, \mathbf{B})$ is computed using modified *joint* inside and outside probabilities

$$\widehat{\beta}_{i:k} = p(\mathbf{A}_{i:k}, \mathbf{B}_{i:k} \,|\, a_{i:k}) \qquad \widehat{\alpha}_{j:k} = p(\mathbf{A}_{0:j}, \mathbf{B}_{0:j}, a_{j:k}, \mathbf{A}_{k:n}, \mathbf{B}_{k:n}) \,, \tag{22}$$

which are constant for the given values of $\mathbf{A}$ and $\mathbf{B}$ and yield, analogously to (16) and (17),

$$p(\mathbf{A}, \mathbf{B}) = \widehat{\beta}_{0:n}\, p_{\mathrm{P}}(a_{0:n}) \qquad \text{and} \qquad \widetilde{p}_{i:k} = \frac{\widehat{\alpha}_{i:k}\, \widehat{\beta}_{i:k}}{p(\mathbf{A}, \mathbf{B})} \,, \tag{23}$$

where $\widetilde{p}_{i:k}$ is the marginal probability of $a_{i:k}$ to exist for this specific assignment of values. As before, the inside probabilities are computed bottom-up and the outside probabilities are computed top-down. MAP estimates are computed by replacing summation or integration with maximisation over the respective variable [13, 33]. For the non-terminal variables, this corresponds to a continuous optimisation problem, which can be efficiently solved using the gradient information provided by libraries with automatic differentiation capabilities, such as `PyTorch` [23]. Unfortunately, due to marginalising over exponentially many network structures, the joint marginal likelihood may be highly non-convex, even for otherwise convex models such as Gaussian RBNs.

2.3 Gaussian RBNs

In a Gaussian RBN (GRBN), the prior, non-terminal, and terminal distributions are linear Gaussians and the termination probability (structural distribution) is constant

$$p_{\mathrm{P}}(a_{0:n}) := \mathcal{N}(a_{0:n}, \mu_{\mathrm{P}}, \Sigma_{\mathrm{P}}) \qquad \text{[prior]} \tag{24}$$

$$p_{\mathrm{N}}(a_{i:j}, a_{j:k} \mid a_{i:k}) := \mathcal{N}(a_{i:j}, a_{i:k}, \Sigma_{\mathrm{NL}})\mathcal{N}(a_{j:k}, a_{i:k}, \Sigma_{\mathrm{NR}}) \qquad \text{[non-terminal]} \tag{25}$$

$$p_{\mathrm{T}}(b_{i+1} \mid a_{i:i+1}) := \mathcal{N}(b_{i+1}, a_{i:i+1}, \Sigma_{\mathrm{T}}) \qquad \text{[terminal]} \tag{26}$$

$$p_{\mathrm{S}}(s{=}\mathtt{T} \mid a) := p_{\mathrm{S}}^{\mathtt{T}} \,. \qquad \text{[termination/structural]} \tag{27}$$

For clarity, we will show all derivations for GRBNs on this basic form. For our evaluations and the application to music, we use a slightly extended version that includes linear transformations, mixtures of Gaussians, and multi-terminal transitions (Section 2.3.1). The derivations do not fundamentally change for the extended case and are detailed in the Appendix.

Adaptive Approximation: If the structure of a GRBN was fixed, all variables would be jointly Gaussian distributed as in a conventional Gaussian Bayesian network (BN) [16]. However, due to the unknown structure, we effectively have a mixture of exponentially many Gaussians, one for each possible structure. While in principle all integrals can be solved analytically, the exponential growth makes exact inference intractable. Therefore, our goal is to derive a parsing strategy that retains tractability by adaptively applying local approximations to the Gaussian mixtures (GMs) that occur in each step.

The base case for both the inside and outside probabilities is a single Gaussian by definition, but the sums in (19) and (21) induce a large number of mixture components in each step. While several methods exist for approximating a given GM by one with fewer components [34, 35], we will here focus on the simplest case of a single Gaussian. The approximation (illustrated in Figure 4, details in the Appendix) can then be efficiently computed in closed form [36, 18]. The inside and outside probabilities are thus approximated with a simple Gaussian

$$\beta_{i:k}(a_{i:k}) \approx c_{i:k}^{(\beta)}\, \mathcal{N}(a_{i:k}, \mu_{i:k}^{(\beta)}, \Sigma_{i:k}^{(\beta)}) \tag{28}$$

$$\alpha_{j:k}(a_{j:k}) \approx c_{j:k}^{(\alpha)}\, \mathcal{N}(a_{j:k}, \mu_{j:k}^{(\alpha)}, \Sigma_{j:k}^{(\alpha)}) \,. \tag{29}$$

Consequently, the marginal posterior distributions over latent variables (17) are also simple Gaussians and the marginal data likelihood (16) can be computed in closed form.

Marginalisation: During inference, we have to integrate over several products of Gaussian distributions to marginalise over the latent variables. To solve these integrals, we make use of the fact that the product of two Gaussians over a variable x can be rewritten as [see e.g. 37]

$$\mathcal{N}(x, \mu_1, \Sigma_1)\,\mathcal{N}(x, \mu_2, \Sigma_2) = \mathcal{N}(\mu_1, \mu_2, \Sigma_1 + \Sigma_2)\,\mathcal{N}(x, \bar{\mu}, \bar{\Sigma}) \tag{30}$$

with

$$\bar{\Sigma} := (\Sigma_1^{-1} + \Sigma_2^{-1})^{-1} \qquad \text{and} \qquad \bar{\mu} := \bar{\Sigma}\left(\Sigma_1^{-1}\mu_1 + \Sigma_2^{-1}\mu_2\right) \,. \tag{31}$$

Hence, when integrating over x, only the first term on the rhs. of (30) remains. A detailed step-by-step derivation of all results can be found in the Appendix. With the latent variables being marginalised out, (19) and (21) become simple mixtures of Gaussians that can be easily approximated to retain the simple analytic form of the inside and outside probabilities.

Tree Induction: Exact tree induction would require joint maximisation of the non-terminal variables and the structure. While in principle, this can be done via gradient descent, in practice this procedure is quickly getting trapped in local optima. Also, the approximation scheme for marginalisation cannot be employed because the max of a Gaussian mixture cannot be easily approximated by a Gaussian (see Figure 4). We therefore compute for each node the split that maximises the marginal likelihood while computing inside probabilities and then select the best tree top-down.

2.3.1 Gaussian RBNs for Music

For the application to music, we slightly extend the basic GRBN discussed so far by introducing *transpositions* and *multi-terminal transitions* (changes in the equations highlighted in blue). The corresponding graphical model of the RBN cell is shown in Figure 5. Furthermore, we describe how GRBNs can be applied to *categorical* data.

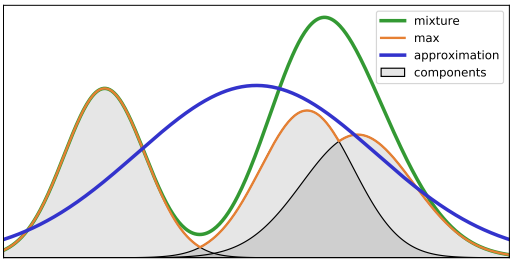

Figure 4: A Gaussian mixture with three components, the resulting mixture distribution (green), maximum (orange) and moment-matching single Gaussian approximation (blue).

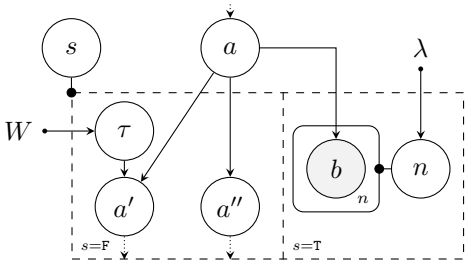

Figure 5: Graphical model of the Gaussian RBN for modelling music. The additional transposition variable τ is marginalised out during inference; the number of jointly generated observations n is uniquely determined by the location in the parse chart.

Transpositions: A transposition rotates the dimensions of the latent variable by a number of steps τ before generating the child. This is achieved by multiplying with an orthonormal transposition matrix T_τ that corresponds to the identity matrix with cyclicly rearranged columns. For the prior distribution, we assume a uniform weighting of all possible transpositions

$$p_{\mathrm{P}}(a_{0:n}) := \sum_{\tau=0}^{D-1} \frac{1}{D} \mathcal{N}(a_{0:n}, T_\tau\, \mu_{\mathrm{p}}, \Sigma_{\mathrm{P}})\,, \qquad \text{[prior]} \tag{32}$$

where D is the dimensionality of the data ($D = 12$ for music in 12-tone equal temperament). For the non-terminal transitions, the probability for a specific transposition is determined by the weight parameter W

$$p_{\mathrm{N}}(a_{i:j}, a_{j:k} \,|\, a_{i:k}) := \sum_{\tau=0}^{D-1} p(\tau \,|\, W)\, \mathcal{N}(a_{i:j}, T_\tau\, a_{i:k}, \Sigma_{\mathrm{NL}}) \mathcal{N}(a_{j:k}, a_{i:k}, \Sigma_{\mathrm{NR}})\,. \tag{33}$$

Note that transpositions are only applied to the left child, because Western classical music is thought to be fundamentally goal directed [2, 8, 38]. This means that the character of a section is largely determined by how it ends (the right child), which should also be reflected in the value of the parent node. In contrast, the role of the left child is to harmonically prepare the ending (or prepare a preparation to the ending etc). We therefore allow for arbitrary transpositions in the left child and we will see below that our model indeed captures the most important type of preparation in Western classical music: the cadential dominant-tonic progression.

Multi-Terminal Transitions: A multi-terminal transition generates multiple observed variables from a single latent variable. The variables are generated i.i.d. and their number is governed by a Poisson distribution with rate parameter λ

$$p_{\mathrm{T}}(b_{i:k} \,|\, a_{i:k}) := \mathrm{Pois}(k - i - 1 \,|\, \lambda) \prod_{j=i+1}^{k} \mathcal{N}(b_j, a_{i:k}, \Sigma_{\mathrm{T}})\,. \qquad \text{[multi-terminal]} \tag{34}$$

Multi-terminal transitions do not conform to the CNF assumed so far and we need to add the term

$$\beta_{i:k}(a_{i:k}) = \cdots + p_{\mathrm{S}}(s_{i:k}{=}\mathtt{T} \,|\, a_{i:k})\, p_{\mathrm{T}}(b_{i:k} \,|\, a_{i:k}) \tag{35}$$

$$= \cdots + p_{\mathrm{S}}^{\mathtt{T}}\, p_{\mathrm{T}}(b_{i:k} \,|\, a_{i:k}) \qquad \text{[for GRBNs, see (27)]} \tag{36}$$

to (19) to account for the possibility to terminate from a higher-level variable. For $k = i + 1$, this term becomes the base case (18) of an RBN in CNF.

Multi-terminal transitions account for the situation where changes in the hierarchical structure occur at a lower rate than the time series was sampled. In between the structural changes, the data are assumed to be generated from the same model, which could also be more elaborate than i.i.d. samples, as long as the relevant model parameters are captured by the RBN's latent variables (here the mean of the Gaussian).

Categorical Data: The observed variables of a GRBN are unconstrained real-valued, which poses a problem if the data are categorical. This situation is comparable to using Gaussian processes (GPs) [39] for classification and can be approached with similar methods. In our application to musical data, we observe one or more notes being played at any particular time and normalise these counts to obtain observations that correspond to the parameter of a categorical distribution. The natural likelihood function for this type of observations is a Dirichlet distribution. Therefore, we adapt the approach suggested in [40] for GPs, who assume a Dirichlet likelihood, which is then approximated by a Gaussian likelihood in log-space. Since an observation from a Dirichlet distribution corresponds to a normalised sample from independent Gamma distributions, each Gamma distribution can be separately approximated by a log-normal distribution, which results in a diagonal covariance matrix for the Gaussian likelihood in log-space. Matching the first and second moment yields [40]

$$\widetilde{b}_j^{(l)} = \log b_j^{(l)} - \widetilde{\Sigma}_{ll}^{(j)}/2 \qquad \text{and} \qquad \widetilde{\Sigma}_{ll}^{(j)} = \log(1/b_j^{(l)} + 1)\ , \tag{37}$$

where $0 < b_j^{(l)} < 1$ is the lth element (normalised count) of the jth observation, $\widetilde{b}_j^{(l)}$ is the corresponding mean of the approximate Gaussian likelihood in log-space, and $\widetilde{\Sigma}_{ll}^{(j)}$ is the lth element on the diagonal of the covariance matrix for the jth observation. We thus have to replace $\widetilde{b}_j$ and $\widetilde{\Sigma}^{(j)}$ for b_j and Σ_{T} in (34).

3 Experiments

We performed a quantitative evaluation on synthetic data as well as hierarchical music analysis on Bach preludes. We show that RBNs are superior to change point detection (CPD) and hierarchical clustering (HC) for tree induction and our method and is able to infer fundamental harmonic principles of Western classical music. Experiments were run on a 3.6 GHz Quad-Core Intel Core i7 processor with 32GB RAM. The triangular plots follow the idea of *scape plots* [41, 42, 43].

3.1 Quantitative Evaluation on Tree Induction

We performed a quantitative evaluation on synthetic data for the task of segmenting a noisy time series and inferring the underlying tree. For comparison, we used the best-performing change point detection (CPD) method from the `ruptures` library [45] for segmenting the time series, combined with bottom-up hierarchical clustering (HC) for inferring the tree structure ("HC/CPD"). For details on methodology, see Appendix.

The evaluation results (Figure 6(b)) show that the full tree estimates from the RBN are clearly superior to those computed by combining CPD and HC, both in terms of precision and recall (and thus also in F1 measure). The marginal node probabilities show an interesting performance pattern. In terms of precision, they surpass the full tree estimates, which means that a node with high marginal probability is very likely to actually exist in the tree (low false-positive rate). However, in terms of recall, the marginal probabilities fall far below the baseline, which means that a node with low marginal probability may in fact occur in the tree (high false-negative rate).

We think that the poor recall measure of the marginal probabilities is primarily due to (and the downside of) a fully Bayesian treatment that quantifies uncertainty. Even if the marginal probabilities have a maximum at the correct node location, probability mass will still spread around it and be allocated to a number of less probable locations. While this is the desired behaviour of a Bayesian method, it inevitably results in a lower recall value. The high precision value confirms that uncertainty is adequately quantified and not underestimated. That being said, the marginal probabilities provide an exceptionally rich basis for qualitative analyses. For instance, looking at Figure 6(a), we see that all ground-truth nodes are located at local maxima of the marginal probabilities. Moreover, we can read off a number other potential node locations, which essentially trace out the grid defined by the piece-wise constant segments. On the synthetic data and give the low noise level, this pattern is very clear and simple, but we will see a more nuanced example on the musical data.

3.2 Hierarchical Music Analysis

Harmonies in Western classical music exhibit a nested hierarchical structure that can be modeled by PCFGs operating on abstract chord symbols [8, 46, 10, 3]. While these grammars can be applied

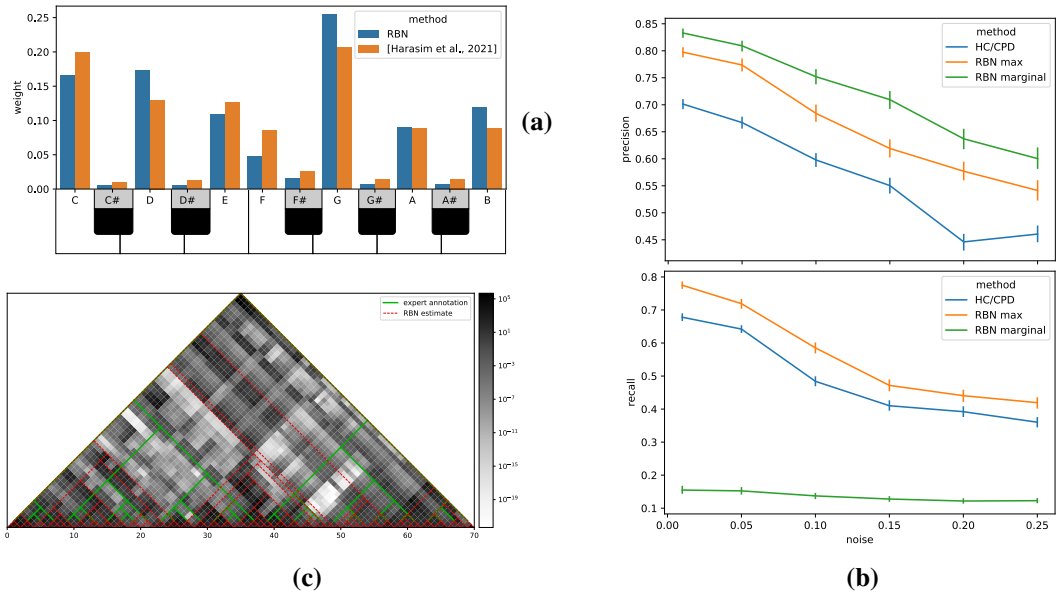

Figure 6: **(a):** The prior mean learned by our GRBN corresponds to the pitch profile of a major scale and is in excellent agreement with most recent Bayesian estimates from the literature [44]. **(b):** Precision and recall w.r.t. the ground-truth trees of 500 sequences for each noise level; error bars indicate 95% confidence intervals from bootstrapping. The RBN tree estimate consistently outperforms the one from HC/CPD, while the marginal estimates from the RBN excel at precision but severely underestimate the overall node probabilities (see text for details). **(c):** On Johann Sebastian Bach's Prelude No. 1 in C major, BWV 846, our model correctly identifies the regular measure-wise grouping at the bottom and identifies a number of reasonable partitions. However, compared to the expert analysis is does not correctly infer the large-scale grouping in to two main sections.

based on expert annotations of a musical score, hierarchical music analysis from the raw note level is an unsolved problem. We trained a GRBN (Section 2.3.1) on the 24 major preludes of Johann Sebastian Bach's "Wohltemperiertes Klavier I & II" (see Appendix for technical details and complete results).

Our first major finding is that the prior mean (Figure 6(a)) corresponds to a major pitch profile as could be expected based on the training data. Second the only two transpositions with non-zero weights are the identity with a weight of 78% and the fifth scale degree (7 semitones) with a weight of 22%. This corresponds to the left child being generated as the dominant of the parent and realises the most important harmonic preparation in Western classical music: the cadential dominant-tonic relation. While the estimated latent tree structure (Figure 6(c)) shows considerable deviations from the expert harmonic analysis, our model was still able to capture fundamental properties of Western classical music, based on only 24 pieces. We therefore think Gaussian RBN are a highly promising approach for hierarchical music analysis from the raw note level.

4 Conclusion

We introduced Recursive Bayesian Networks (RBNs), a novel class of probabilistic models that unifies the strengths of probabilistic context-free grammars (PCFGs) and dynamic Bayesian networks (DBNs), generalising both model classes. We defined RBNs as a weighted mixture over tree-structured Bayesian networks and described how to perform inference over both the model structure and the discrete or continuous variables by leveraging parsing methods for PCFGs. The provided formalisation connects with the methods for formals grammar as well as with the versatile notation for graphical models. On two data sets, we demonstrated the potential of RBNs for modelling nested hierarchical dependencies in real-valued time series.

The class of RBNs represents a fundamental contribution to the machine learning toolkit by unifying two of the most important approaches for modelling sequential data.

References

[1] Dan Jurafsky. *Speech & Language Processing*. Pearson Education India, 2000.

[2] Fred Lerdahl and Ray Jackendoff. *A Generative Theory of Tonal Music*. MIT press, 1983.

[3] Martin Rohrmeier. The Syntax of Jazz Harmony: Diatonic Tonality, Phrase Structure, and Form. *Music Theory and Analysis (MTA)*, 7(1):1–63, April 2020.

[4] Andrew G. Barto and Sridhar Mahadevan. Recent Advances in Hierarchical Reinforcement Learning. *Discrete Event Dynamic Systems*, 13(4):341–379, 2003.

[5] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning and Acting*. Cambridge University Press, 2016.

[6] Dick Grune and Ceriel JH Jacobs. Parsing techniques. *Monographs in Computer Science. Springer,*, page 13, 2007.

[7] Christopher W. Geib and Robert P. Goldman. A probabilistic plan recognition algorithm based on plan tree grammars. *Artificial Intelligence*, 173(11):1101–1132, July 2009.

[8] Martin Rohrmeier. Towards a generative syntax of tonal harmony. *Journal of Mathematics and Music*, 5(1):35–53, March 2011.

[9] Florent Jacquemard, Pierre Donat-Bouillud, and Jean Bresson. A structural theory of rhythm notation based on tree representations and term rewriting. In *International Conference on Mathematics and Computation in Music*, pages 3–15. Springer, 2015.

[10] Daniel Harasim, Martin Rohrmeier, and Timothy J. O'Donnell. A Generalized Parsing Framework for Generative Models of Harmonic Syntax. In *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pages 152–159, Paris, 2018.

[11] Tadao Kasami. An efficient recognition and syntax-analysis algorithm for context-free languages. *Coordinated Science Laboratory Report no. R-257*, 1966.

[12] Daniel H Younger. Recognition and parsing of context-free languages in time n3. *Information and control*, 10(2):189–208, 1967.

[13] Joshua Goodman. Semiring parsing. *Computational Linguistics*, 25(4):573–605, 1999.

[14] Christopher Manning and Hinrich Schutze. *Foundations of Statistical Natural Language Processing*. MIT press, 1999.

[15] Kevin Patrick Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California, Berkeley, 2002.

[16] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT press, 2009.

[17] Greg Welch and Gary Bishop. An introduction to the Kalman filter. Technical report, University of North Carolina, 1995.

[18] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1st ed. 2006. corr. 2nd printing edition, 2007.

[19] Frank Drewes, H.-J. Kreowski, and Annegret Habel. Hyperedge replacement graph grammars. In *Handbook Of Graph Grammars And Computing By Graph Transformation: Volume 1: Foundations*, pages 95–162. World Scientific, 1997.

[20] Joost Engelfriet and Grzegorz Rozenberg. Node replacement graph grammars. In *Handbook Of Graph Grammars And Computing By Graph Transformation: Volume 1: Foundations*, pages 1–94. World Scientific, 1997.

[21] Eric J. Golin. Parsing visual languages with picture layout grammars. *Journal of Visual Languages & Computing*, 2(4):371–393, December 1991.

[22] Grzegorz Rozenberg. *Handbook of Graph Grammars and Computing by Graph Transformation*, volume 1. World scientific, 1997.

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[24] Jason Eisner. Inside-Outside and Forward-Backward Algorithms Are Just Backprop (tutorial paper). In *Proceedings of the Workshop on Structured Prediction for NLP*, pages 1–17, Austin, TX, 2016. Association for Computational Linguistics.

[25] Yoon Kim, Chris Dyer, and Alexander Rush. Compound Probabilistic Context-Free Grammars for Grammar Induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2369–2385, Florence, Italy, 2019. Association for Computational Linguistics.

[26] Yu Zhang, Houquan Zhou, and Zhenghua Li. Fast and accurate neural CRF constituency parsing. In *Proceedings of IJCAI*, pages 4046–4053, 2020.

[27] Alexander M. Rush. Torch-Struct: Deep Structured Prediction Library. *arXiv:2002.00876 [cs, stat]*, February 2020.

[28] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, 2013.

[29] Daniel Eaton and Kevin Murphy. Bayesian structure learning using dynamic programming and MCMC. *arXiv:1206.5247 [cs, stat]*, June 2012.

[30] Mathias Drton and Marloes H. Maathuis. Structure Learning in Graphical Modeling. *Annual Review of Statistics and Its Application*, 4(1):365–393, March 2017.

[31] Tom Minka and John Winn. Gates. In *Advances in Neural Information Processing Systems*, pages 1073–1080, 2009.

[32] Brendan J. Frey. Extending factor graphs so as to unify directed and undirected graphical models. In Christopher Meek and Uffe Kjærulff, editors, *UAI '03, Proceedings of the 19th Conference in Uncertainty in Artificial Intelligence, Acapulco, Mexico, August 7-10 2003*, pages 257–264. Morgan Kaufmann, 2003.

[33] Liang Huang and David Chiang. Better k-best parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology - Parsing '05*, pages 53–64, Vancouver, British Columbia, Canada, 2005. Association for Computational Linguistics.

[34] Marco F. Huber and Uwe D. Hanebeck. Progressive Gaussian mixture reduction. In *2008 11th International Conference on Information Fusion*, pages 1–8, June 2008.

[35] David F. Crouse, Peter Willett, Krishna Pattipati, and Lennart Svensson. A look at Gaussian mixture reduction algorithms. In *14th International Conference on Information Fusion*, pages 1–8. IEEE, 2011.

[36] U. Orguner and M. Demırekler. Analysis of single Gaussian approximation of Gaussian mixtures in Bayesian filtering applied to mixed multiple-model estimation. *International Journal of Control*, 80(6):952–967, June 2007.

[37] Kaare Brandt Petersen, Michael Syskind Pedersen, Jan Larsen, Korbinian Strimmer, Lars Christiansen, Kai Hansen, Liguo He, Loic Thibaut, Miguel Barão, Stephan Hattinger, Vasile Sima, and We The. The matrix cookbook. Technical report, 2006.

[38] Stefan Koelsch, Martin Rohrmeier, R. Torrecuso, and S. Jentschke. Processing of hierarchical syntactic structure in music. *Proceedings of the National Academy of Sciences*, 110(38):15443–15448, September 2013.

[39] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, Massachusetts / London, England, 2006.

[40] Dimitrios Milios, Raffaello Camoriano, Pietro Michiardi, Lorenzo Rosasco, and Maurizio Filippone. Dirichlet-based Gaussian processes for large-scale calibrated classification. In *Advances in Neural Information Processing Systems*, pages 6005–6015, 2018.

[41] Craig Stuart Sapp. Harmonic Visualizations of Tonal Music. In *ICMC*, volume 1, pages 419–422. Citeseer, 2001.

[42] Meinard Müller and Nanzhu Jiang. A Scape Plot Representation for Visualizing Repetitive Structures of Music Recordings. In *ISMIR*, pages 97–102. Citeseer, 2012.

[43] Robert Lieck and Martin Rohrmeier. Modelling Hierarchical Key Structure With Pitch Scapes. In *Proceedings of the 21st International Society for Music Information Retrieval Conference*, pages 811–818, Montréal, Canada, 2020.

[44] Daniel Harasim, Fabian C. Moss, Matthias Ramirez, and Martin Rohrmeier. Exploring the foundations of tonality: Statistical cognitive modeling of modes in the history of Western classical music. *Humanities and Social Sciences Communications*, 8(1):1–11, January 2021.

[45] Charles Truong, Laurent Oudre, and Nicolas Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, February 2020.

[46] Martin Rohrmeier and Marcus Pearce. Musical Syntax I: Theoretical Perspectives. In Rolf Bader, editor, *Springer Handbook of Systematic Musicology*, pages 473–486. Springer Berlin Heidelberg, Berlin, Heidelberg, 2018.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section...
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes]
(b) Did you mention the license of the assets? [Yes]
(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]