Unifying Sum-Product Networks and Submodular Fields

Abram L. Friesen [1] **Pedro Domingos** [1]

Abstract

Sum-product networks (SPNs) are a relatively new class of deep probabilistic models that are expressive but also tractable, due in part to their ability to exploit local structure in the form of context-specific independence and determinism. However, there remain many types of problems that are highly structured, but for which SPNs are intractable. In particular, the MAP solution of a Markov random field (MRF) with submodular energy can be computed in low-order polynomial time, but an SPN representing the same distribution would be intractable. In this work, we present submodular sum-product networks (SSPNs), a generalization of SPNs in which sum-node weights can be defined by a submodular energy function. SSPNs combine the expressivity and depth of SPNs with the ability to efficiently compute the MAP state of a combinatorial number of labelings afforded by submodularity, greatly increasing the expressivity of the SPN model class. We develop a move-making algorithm for computing the (approximate) MAP state of an SSPN and analytically show that it is both efficient and convergent. Empirically, we show exponential improvements in inference time compared to traditional inference algorithms such as α-expansion and belief propagation, while returning comparable minima. Finally, we present promising empirical results when using SSPNs for scene understanding.

1. Introduction

In recent years, there has been a growing interest in probabilistic models that are expressive but still permit tractable inference. An important subset of this space can be characterized as sum-product networks (SPNs) (Poon & Domingos, 2011; Gens & Domingos, 2012), an expressive class of deep probabilistic models that consist of many layers of hidden variables and can have unbounded treewidth. Despite this depth and corresponding expressivity, exact marginal and MPE (with respect to both the hidden and observed variables) inference in an SPN is guaranteed to take time linear in its size, whereas more traditional graphical models, such as Markov random fields (MRFs), have inference complexity that is both best- and worst-case exponential in their treewidth (Chandrasekaran et al., 2008). Much of this exponential improvement in inference complexity is due to the ability of SPNs to exploit local structure in the form of context-specific independence (Boutilier et al., 1996) and determinism in the underlying distribution.

However, submodularity, another form of local structure that permits exponential reductions in inference complexity and is commonly exploited to build tractable models in fields from computer vision (Greig et al., 1989; Boykov et al., 2001; Kolmogorov & Rother, 2007; Komodakis et al., 2007) to social network modeling (Kempe et al., 2003; Mossel & Roch, 2007), is not a property that SPNs are able to take advantage of, despite its clear benefits. For example, scene understanding (or semantic segmentation) is commonly formulated as a pairwise MRF with a node for each pixel in the image and a label for each semantic class. In the general case, finding the optimal segmentation of an image is intractable. However, if the MRF is submodular (alternatively, regular or attractive) (Kolmogorov & Zabih, 2004), meaning that each pixel prefers to have the same label as its neighbors, then the exact MAP solution of a binary MRF can be recovered in low-order polynomial time in the number of pixels with the use of a graph-cut algorithm[1] (Greig et al., 1989; Boykov & Kolmogorov, 2004). For multi-label problems, a constant factor approximation can be recovered with the same time complexity using a move-making algorithm, such as α-expansion (Boykov et al., 2001), which solves a series of binary graph cut problems to find the multi-label solution. However, pairwise MRFs are insufficiently expressive for complex tasks such as scene understanding, as they require a combinatorial number of labels to model higher-level re-

[1]Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle, WA USA. Correspondence to: Abram Friesen <afriesen@cs.washington.edu>, Pedro Domingos <pedrod@cs.washington.edu>.

Presented at the ICML 2017 Workshop on Principled Approaches to Deep Learning, Sydney, Australia, 2017.

[1]Formally, a min-cut/max-flow algorithm (Ahuja et al., 1993) on a graph constructed from the MRF.

lationships, such as constituency (part-subpart) or subcategorization (superclass-subclass) between arbitrary regions of the image, which is intractable. Conversely, an SPN can efficiently model these types of high-level relationships but cannot efficiently represent the possible segmentations of an image.

In this work, we present *submodular sum-product networks* (SSPNs), a novel probabilistic model that generalizes both SPNs and submodular MRFs, thereby combining the expressive power of SPNs with the tractable inference properties of submodular random fields. An SSPN is a sum-product network in which the weight of each child of a sum node is given by the energy of a particular state of a submodular energy function. In the case of scene understanding, an SSPN over an image can be interpreted as an instantiation of all possible parse trees of that image with respect to a given image grammar, where the probability distribution over the segmentations of a production on a particular region is defined by a submodular random field over the pixels in that region. Importantly, SSPNs permit objects and regions to take *arbitrary shapes*, instead of restricting the set of possible shapes as has previously been necessary for tractable inference. By exploiting submodularity, we develop a highly-efficient approximate inference algorithm, INFERSSPN, for computing the MAP state of the SSPN (equivalently, the optimal parse of the image). INFERSSPN is an iterative move-making-style algorithm that provably converges to a local minimum of the energy, reduces to α-expansion in the case of a trivial grammar, and has worst-case complexity $O(|G|c(n)n)$ for each iteration (but in practice behaves as $O(|G|c(n))$), where n is the number of pixels, $c(n)$ is the complexity of a single graph cut (linear or quadratic in n) and $|G|$ is the size of the grammar. As with other move-making algorithms, INFERSSPN converges to a local minimum with respect to an exponentially-large set of neighbors, overcoming many of the main issues of local minima (Boykov et al., 2001). Empirically, we compare INFERSSPN to belief propagation (BP) and to α-expansion. We show that INFERSSPN parses images in exponentially less time than both of these while returning energies comparable to α-expansion, which is itself guaranteed to return energies within a constant factor of the true optimum. We also show promising results on scene understanding using oracle-induced grammars.

The literature on using higher-level relationships for scene understanding is vast. We briefly discuss the most relevant work on hierarchical random fields over multiple labels, image grammars for segmentation, and neural parsing methods. Hierarchical random field models (e.g., Russell et al. (2010); Lempitsky et al. (2011)) define MRFs with multiple layers of hidden variables and then perform inference, often using graph cuts to efficiently extract the MAP solution. However, these models are typically restricted to just a few layers and to pre-computed segmentations of the image, and thus do not allow arbitrary region shapes. In addition, they require a combinatorial number of labels to encode complex grammar structures. Previous grammar-based methods for scene understanding, such as Zhu & Mumford (2006) and Zhao & Zhu (2011), have used MRFs with AND-OR graphs (Dechter & Mateescu, 2007), but needed to restrict their grammars to a very limited set of productions and region shapes in order to perform inference in reasonable time, and are thus much less expressive than SSPNs. Finally, neural parsing methods such as those in Socher et al. (2011) and Sharma et al. (2014) use recursive neural network architectures over superpixel-based features to segment an image; thus, these methods also do not allow arbitrary region shapes. Further, Socher et al. (2011) and Sharma et al. (2014) respectively create parse trees greedily and randomly, whereas INFERSSPN finds the approximately optimal parse tree.

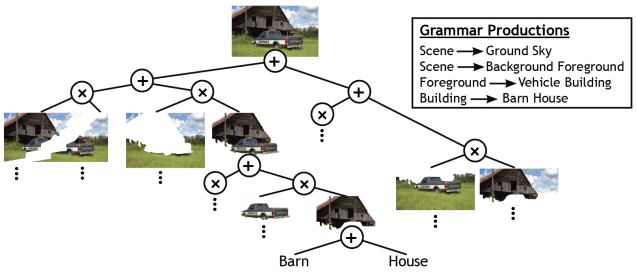


Figure 1. A partial (submodular) sum-product network for parsing an image with respect to the grammar shown. There is a sum node for each nonterminal symbol with a child sum node for each production of that symbol. Each sum node for a production has a child product node for each possible segmentation of its region.

2. Submodular sum-product networks

A sum-product network (SPN) (Poon & Domingos, 2011; Gens & Domingos, 2013) is a deep, probabilistic model typically represented as a directed acyclic graph (DAG) of sum nodes, product nodes, and leaf functions, with weights on the child-edges of each sum node. Product nodes in an SPN are decomposable, meaning that the children of a product node must have disjoint scopes, where a node's scope is the set of variables its descendant leaf functions are over. Decomposability ensures that inference in SPNs can be performed exactly in time linear in the size of the network, but is a weak restriction in that SPNs can still efficiently represent distributions with high treewidth (Friesen & Domingos, 2016). Without loss of generality, we assume that sums and products are arranged in alternating layers. See the cited papers for more details on SPNs.

For the purposes of this paper, we define an SPN using a very expressive grammar containing a symbol for each sum node and a production for each product node. Each production produces the symbols corresponding to its children and the production probabilities are defined by the edge weights. Due to decomposability, the scopes of the symbols produced by a production partition the scope of the parent symbol. SPNs strictly subsume standard grammar formulations, such as probabilistic context-free grammars (PCFGs) (Jurafsky & Martin, 2000), since they allow partitions to have non-uniform probabilities and to be non-contiguous. Despite this flexibility, however, SPNs cannot fully exploit it, because the number of possible partitions of n variables is exponential in n, and an SPN would need exponential size to represent these. For example, Poon & Domingos (2011) define an SPN over images but can only retain tractability by restricting symbols to have rectangular regions (scopes). However, objects in images are rarely rectangular, indicating a truth about many data types: the partitioning (decomposability) structure is in general neither known a priori nor highly regular.

In this work, we present submodular sum-product networks (SSPNs), a generalization of SPNs that uses submodular Markov random fields (MRFs) to define the probability of each partition of a sum node's scope and efficiently encode this exponentially-large set. Extending the grammar formulation, a symbol in an SSPN maps to an exponentially-large set of sum nodes in some implicit underlying SPN, where each such sum node has a different scope. Productions similarly encode all possible partitions of each symbol's scope and similarly map to (implicit) exponentially-large sets of product nodes. The supplement [2] contains more details on this mapping. With respect to images, an SSPN encodes a grammar in which each symbol can have arbitrary region shape at inference time, providing a much more flexible representation of the components of an image and their relationships. It is not necessary to define SSPNs with respect to a grammar or an image but we do so here with respect to scene understanding and a flexible grammar (that subsumes SPNs) to simplify the exposition.

2.1. Model definition

As with SPNs, an SSPN defines a generative model, the structure of which can be defined by a non-recursive stochastic grammar $G = (N, \Sigma, R, S)$, where N is a finite set of nonterminal symbols; Σ is a finite set of terminal symbols; R is a finite set of productions $R = \{v : X \to Y_1 \dots Y_k\}$ with head symbol $X \in N$ and constituent symbols $Y_i \in N \cup \Sigma$ for $i = 1 \dots k$ and $k > 0$; and $S \in N$ is the start symbol, which does not appear on the right-hand side of any production. For scene understanding, an SSPN defines a generative model of an image. A parse (tree) $t \in \mathcal{T}_G$ of an image with respect to the SSPN defined by grammar G specifies a hierarchy of pairs of productions and their regions $(v, \mathcal{R})$, where each region specifies a subset of the pixels, can have arbitrary shape, and is a subset of the regions of its ancestors. An example of an SSPN for a farm scene is shown in Figure 1. Given the starting symbol S and the region containing the entire image, the generative process is to choose a production of that symbol and then partition (segment) the region into disjoint subregions – one for each constituent of the production. This process recurses on each subregion-constituent pair, and terminates when the constituent is a terminal symbol, at which point the pixels for that region are generated. Productions are sampled from each symbol's mixture distribution over its productions and segmentations of a given region are sampled from a (submodular) MRF over that region. Specifically, each production has a corresponding MRF defined over the entire image. The probability of a segmentation of any subregion is given by the sub-MRF containing only those pixels and edges in that subregion.

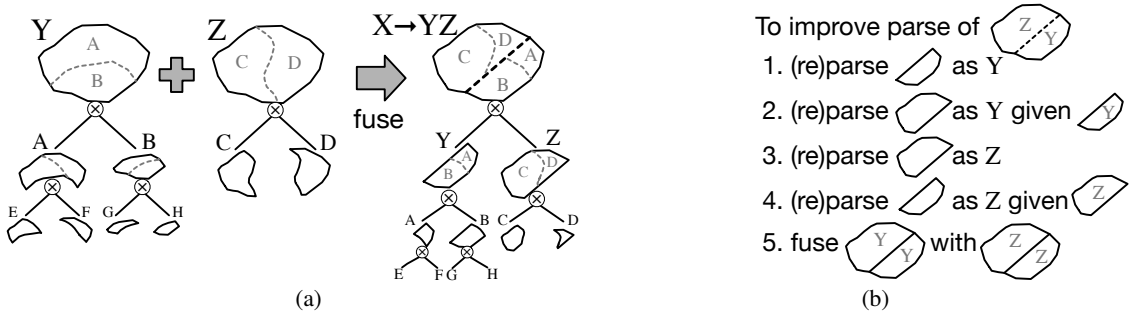
We denote the labeling corresponding to the segmentation of pixels in a region $\mathcal{R}$ for production $v : X \to Y_1 \dots Y_k$ as $\mathbf{y}^v \in \mathcal{Y}_v^{|\mathcal{R}|}$, where $\mathcal{Y}_v = \{Y_1, \dots, Y_k\}$. The region of a constituent Y_i is the set of pixels in $\mathcal{R}$ with that label, i.e., $\mathcal{R}_{Y_i} = \{p \in \mathcal{R} : y_p^v = Y_i\}$. The probability of image $\mathcal{I}$ is $p_{\mathbf{w}}(\mathcal{I}) = \sum_{t \in \mathcal{T}_G} p_{\mathbf{w}}(t, \mathcal{I})$, where the joint probability is

$$p_{\mathbf{w}}(t, \mathcal{I}) = \frac{1}{Z} \prod_{(v,\mathcal{R}) \in t} \widetilde{p}_{\mathbf{w}}(v|\text{head}(v)) \cdot \widetilde{p}_{\mathbf{w}}(\mathbf{y}^v, \mathcal{I}|v, \mathcal{R}) \propto \exp\Big\{ -\sum_{(v,\mathcal{R}) \in t} w_v + E_{\mathbf{w}}^v(\mathbf{y}^v, \mathcal{I}, \mathcal{R}) \Big\}, \quad (1)$$

which we will also write as $p_{\mathbf{w}}(t, \mathcal{I}) \propto \exp(-E_{\mathbf{w}}(t, \mathcal{I}))$, and where $\widetilde{p}$ are unnormalized probabilities, $\mathbf{w}$ are the model parameters, w_v is the production cost associated with the mixture model of its head symbol, E is the energy function, and $Z = \sum_{t \in \mathcal{T}_G} \exp(-E_{\mathbf{w}}(t, \mathcal{I}))$ is the partition function. To simplify notation, we omit $v, \mathcal{I}$, and $\mathcal{R}$ when clear from context and sum over just v.

The energy of a segmentation of a production v on region $\mathcal{R}$ is given by a pairwise MRF as $E(\mathbf{y}^v, \mathcal{R}) = \sum_{p \in \mathcal{R}} \theta_p^v(y_p^v; \mathbf{w}) + \sum_{(p,q) \in \mathcal{E}} \theta_{pq}^v(y_p^v, y_q^v; \mathbf{w})$, where θ_p^v and θ_{pq}^v are the unary and pairwise costs, and $\mathcal{E}$ are the edges in $\mathcal{R}$. These MRFs can be parameterized arbitrarily, but in order to permit efficient inference, we require that θ_{pq}^v satisfies the submodularity condition $\theta_{pq}^v(Y_1, Y_1) + \theta_{pq}^v(Y_2, Y_2) \leq \theta_{pq}^v(Y_1, Y_2) + \theta_{pq}^v(Y_2, Y_1)$ for all productions $v : X \to Y_1 Y_2$, once the grammar has been converted to a grammar in which each production has only two constituents, which is always possible and in the worst case increases the grammar size quadratically (Jurafsky & Martin,

[2] http://homes.cs.washington.edu/~pedrod/papers/padl17sp.pdf



(a) (b)

Figure 2. The two main components of INFERSSPN: (a) Parsing a region $\mathcal{R}$ as $X \to YZ$ by fusing two parses of $\mathcal{R}$ as $Y \to AB$ and as $Z \to CD$, and (b) Improving the parse of $\mathcal{R}$ as $X \to YZ$ by (re)parsing each of its subregions, taking the union of the new Y and Z parses of $\mathcal{R}$, and then fusing these new parses.

2000; Chomsky, 1959). We also require for every production $v \in R$, and for every production c that is a descendant of v in the grammar, that $\theta^v_{pq}(y^v_p, y^v_q) \geq \theta^c_{pq}(y^c_p, y^c_q)$ for all possible labelings $(y^v_p, y^v_q, y^c_p, y^c_q)$, where $y^v_p, y^v_q \in \mathcal{Y}_v$ and $y^c_p, y^c_q \in \mathcal{Y}_c$, to ensure that segmentations of higher-level productions are submodular relative to their descendants. This also encodes the reasonable assumption that higher-level abstractions are separated by stronger, shorter boundaries (relative to their size), while lower-level objects are more likely to be composed of smaller, more intricately-shaped regions.

3. Inference

While scene understanding is often thought of as simply labeling each pixel of an image with its semantic class, a true understanding of a scene also requires knowledge of the relationships between image regions at various levels of abstraction, including concepts such as composition and subcategorization. Identifying these structures and relationships for a particular image can be achieved by finding the best parse of an image $\mathcal{I}$ with respect to a grammar G (or, equivalently, performing MAP inference in the corresponding SSPN), i.e., $t^* = \arg\max_{t \in \mathcal{T}_G} p(t|\mathcal{I}) = \arg\min_{t \in \mathcal{T}_G} \sum_{v \in t} E(\mathbf{y}^v, \mathcal{R}_v, \mathcal{I})$. The per-pixel semantic labels can also be recovered from the parse if they are encoded in the grammar.

With a PCFG (Jurafsky & Martin, 2000), the optimal parse of a sentence can be recovered exactly in time cubic in the length of the sentence with the CYK algorithm (Hopcroft & Ullman, 1979), which uses dynamic programming to efficiently parse a sentence in a bottom-up pass through the grammar, forming larger, more abstract symbols as it proceeds. This is possible because each sentence only has a linear number of possible split points; however, this approach is intractable for images and other higher-dimensional data types as there are an exponential number of possible split points. Instead, we first note that if we knew the regions in the optimal parse tree, parsing an SSPN would reduce to parsing a small SPN, which can be accomplished efficiently. Conversely, given the production choices in the optimal parse tree, parsing an SSPN reduces to finding the best labeling of a flat submodular MRF. However, neither the regions nor the productions are known. Thus, our inference algorithm, INFERSSPN, estimates these in an alternating fashion, first estimating the regions of each symbol in a downward pass through the grammar and then iteratively constructing a parse tree for each production of each symbol for each of its regions in an upward pass, choosing the best of these to retain and use when constructing the parses of its ancestors, as in the CYK algorithm.

However, the number of possible partitions for a production is still exponential in the region size. The key to efficiently creating good parses is to exploit submodularity. Recall that each production has a single associated MRF that defines the segmentation probability for any subregion. INFERSSPN similarly constructs for each production a parse of all pixels in the image and uses this to represent the parse of any subregion. Crucially, each parse's distribution is also represented by a submodular energy function, which allows INFERSSPN to take the parses of two symbols Y and Z and *fuse* them to create a parse for the production $v : X \to YZ$ by minimizing a submodular energy function. The resulting parse of v can itself be fused to create new parses. We define this procedure below.

3.1. Parse tree construction

Recall that a parse tree t on region $\mathcal{R}$ has energy $E(t, \mathcal{R}) = \sum_{u \in t} E(\mathbf{y}^u, \mathcal{R}_u)$, which can be written as $E(t, \mathcal{R}) = w(t) + \sum_{p \in \mathcal{R}} \theta^t_p + \sum_{(p,q) \in \mathcal{E}} \theta^t_{pq}$, where $w(t) = \sum_{u \in t} w_u$; $[\cdot]$ is the indicator function; $\theta^t_p = \sum_{u \in t} \theta^u_p(y^u_p) \cdot [p \in \mathcal{R}_u]$; and $\theta^t_{pq} = \sum_{u \in t} \theta^u_{pq}(y^u_p, y^u_q) \cdot [(p, q) \in \mathcal{E}_u]$. This can be represented as a flat MRF over the pixels in $\mathcal{R}$ with an exponential number of labels (one for each parse path in the grammar), but inference (segmentation) in this MRF is hard due to the number of labels and the hard constraints

Algorithm 1 Compute the (approximate) MAP assignment of the SSPN variables (i.e., choose the productions and regions / labelings) for an image and a grammar. This is equivalent to parsing the image.

Input: The image $\mathcal{I}$, a non-recursive grammar $G = (N, \Sigma, R, S, \mathbf{w})$, and an (optional) input parse $\hat{t}$.
Output: A parse of the image, t^*, with energy $E(t^*, \mathcal{I}) \leq E(\hat{t}, \mathcal{I})$.

function INFERSSPN($\mathcal{I}, G, \hat{t}$)
  for each terminal $T \in \Sigma$ **do** $t_{\mathcal{R}_T} \leftarrow$ the trivial parse with all pixels parsed as T
  while the energy of any production of the start symbol S has not converged **do**
    for each node in $\hat{t}$ with production $u : X \rightarrow YZ$, region $\mathcal{R}_X$, and subregions $\mathcal{R}_Y, \mathcal{R}_Z$ **do**
      append $\mathcal{R}_Y, \mathcal{R}_Z$ to region lists $\mathcal{R}[Y], \mathcal{R}[Z]$ and set as the child regions of u for $\mathcal{R}_X$
    for each symbol $X \in N$ with no regions in $\mathcal{R}[X]$ **do** add $\mathcal{R}_X = \mathcal{I}$ to $\mathcal{R}[X]$
    for each symbol $X \in N$, in reverse topological order **do** *// upward pass to parse SSPN*
      for each region $\mathcal{R}_X$ in region list $\mathcal{R}[X]$ **do**
        for each production $v : X \rightarrow YZ$ of symbol X **do**
          $\mathcal{R}_Y, \mathcal{R}_Z \leftarrow$ the child regions of v for $\mathcal{R}_X$ if they exist, else choose heuristically
          $t_v, e_v \leftarrow$ fuse $t_{\mathcal{R}_Y}$ and $t_{\mathcal{R}_Z}$ as production v over region $\mathcal{R}_X$
          $t_{\bar{v}}, e_{\bar{v}} \leftarrow$ fuse $t_{\mathcal{R}_Y}$ and $t_{\mathcal{R}_Z}$ as production v over region $\mathcal{R}_{\overline{X}} = \mathcal{I} \backslash \mathcal{R}_X$ given t_v
        $t_{\mathcal{R}_X}, e_{\mathcal{R}_X} \leftarrow$ the full parse $t_v \cup t_{\bar{v}}$ with lowest energy e_v *// choose best parse of $\mathcal{R}_X$*
    $\hat{t}, \hat{e} \leftarrow t_{\mathcal{R}_S}, e_{\mathcal{R}_S}$ *// S only ever has a single region, which contains all of the pixels*
  return $\hat{t}, \hat{e}$

imposed by the grammar (e.g., each symbol can only be produced once). Instead, if we construct each parse tree one production at a time in a bottom-up fashion then we only need solve a small number of relatively easy segmentation problems. Note, however, that by doing this we've exchanged a single hard but global labeling problem for a series of easy but local labeling problems, so the resulting parse will not in general be globally optimal, although we will see that it is still quite good.

In particular, given a production $v : X \rightarrow Y_1 Y_2$ and parse trees t_1, t_2 over the same region $\mathcal{R}$ and with head symbols Y_1, Y_2, respectively, then for any labeling $\mathbf{y}^v \in \{Y_1, Y_2\}^{|\mathcal{R}|}$ of $\mathcal{R}$ we can construct a third parse tree t_v over region $\mathcal{R}$ with root production v, labeling $\mathbf{y}^v$, and subtrees t_1', t_2' over regions $\mathcal{R}_1, \mathcal{R}_2$, respectively, such that $\mathcal{R}_i = \{p \in \mathcal{R} : y_p^v = Y_i\}$ and $t_i' = t_i \cap \mathcal{R}_i$ for each i, where the intersection of a parse tree and a region $t \cap \mathcal{R}$ is the parse tree resulting from intersecting $\mathcal{R}$ with the region at each node in t. The energy of the resulting parse tree t_v is $E(t_v, \mathcal{R}) = E(v, t_1, t_2, \mathbf{y}, \mathcal{R}) = w_v + w(t_1) + w(t_2) + \sum_{p \in \mathcal{R}} (\theta_p^{t_1} \delta_{y_p Y_1} + \theta_p^{t_2} \delta_{y_p Y_2}) + \sum_{(p,q) \in \mathcal{E}} (\theta_{pq}^{t_1} \delta_{y_p Y_1} \delta_{y_q Y_1} + \theta_{pq}^{t_2} \delta_{y_p Y_2} \delta_{y_q Y_2} + \theta_{pq}^{v}(Y_1, Y_2) \delta_{y_p Y_1} \delta_{y_q Y_2})$, where δ_{ij} is the Kronecker delta. Note that the production costs are constant given v, t_1, and t_2. The quality of t_v depends on the particular labeling (segmentation) $\mathbf{y}^v$ used, where the best parse tree is the one with minimum energy $E(t_v, \mathcal{R})$. We refer to this process of optimally combining the parses of the constituents of a production as *fusion*, as it is analogous to the fusion moves of Lempitsky et al. (2010). Fusion forms the core of INFERSSPN.

Definition 1. *For any production $v : X \rightarrow Y_1, Y_2$ and two parse trees t_1, t_2 over region $\mathcal{R}$ with head symbols Y_1, Y_2, the* fusion *of t_1 and t_2 as v is the parse tree t_X constructed from the minimum energy labeling $\mathbf{y}^v = \arg\min_{\mathbf{y} \in \mathcal{Y}_v^{|\mathcal{R}|}} E(v, t_1, t_2, \mathbf{y})$.*

Figure 2a shows an example of fusing two parse trees to create a new parse tree. Although fusion requires finding the optimal labeling from an exponentially large set, $E(v, t_1, t_2, \mathbf{y})$ is submodular and can be efficiently optimized with a single graph cut. All proofs are presented in the supplement.

Proposition 1. *The energy $E(v, t_1, t_2, \mathbf{y}^v)$ of the fusion of parse trees t_1, t_2 over region $\mathcal{R}$ with head symbols Y_1, Y_2 for a production $v : X \rightarrow Y_1 Y_2$ is submodular.*

By exploiting submodularity, each fusion move finds the (locally) optimal parse of a production from a combinatorial number of possible parses; i.e., all parses that can be constructed from choosing, for each pixel in the region, the label specified in one of the sub-trees. Once a parse tree has been constructed, INFERSSPN improves that parse tree on subsequent iterations, as shown in Figure 2b. In particular, for a parse tree t with a subtree t_i, then any improvement to the energy of t_i will improve the energy of t. We show this more formally in the supplement. Finally, we define the union $t = t_1 \cup t_2$ of two parse trees that have the same production choices but disjoint regions, as the parse tree t with the same production choices and in which the region of each node in t is the union of the regions of its corresponding nodes in t_1 and t_2.

3.2. INFERSSPN

Pseudocode for our algorithm, INFERSSPN, is presented in Algorithm 1. INFERSSPN is an iterative bottom-up algorithm based on graph cuts (Kolmogorov & Zabih, 2004) that efficiently and provably converges to a local minimum of the energy function. Each iteration of INFERSSPN improves the current parse $\hat{t}$ in a flexible manner that allows any of its productions or regions (equivalently, labelings) to change.

If an initial parse $\hat{t}$ is provided, INFERSSPN first determines the regions for each symbol and their hierarchical relationships (line 5) in a downward pass through $\hat{t}$. If not, then each symbol is assigned the region containing the entire image (line 6), which serves as a powerful initialization method. INFERSSPN then iteratively constructs a parse for each production of each symbol for each of its regions in a single upward pass through the grammar from the terminals to the start symbol (line 7). The parse of a production $v : X \rightarrow YZ$ for region $\mathcal{R}_X$ is constructed on line 11 by fusing the previously-computed parses of Y and Z for the child regions of $\mathcal{R}_X$ (i.e., its subregions in $\hat{t}$). If $\mathcal{R}_X$ has no children, then it was not in $\hat{t}$ and we can use the parses of any region of Y and Z (since they parse the entire image). Most symbols only have a single region, so no choice is required, but in the rare case of multiple regions, one can be chosen by estimating a bound on its energy or even randomly, as this choice does not affect convergence. In our experiments, we also set the region of each symbol that is not in $\hat{t}$ as the region of its closest ancestor in $\hat{t}$. INFERSSPN then parses the remainder of the image $\mathcal{R}_{\overline{X}}$ as v given the parse of $\mathcal{R}_X$ as v, meaning that the $\theta^v_{pq}(y^v_p, y^v_q)$ terms for edges between pixels $p \in \mathcal{R}_{\overline{X}}$ and $q \in \mathcal{R}_X$ are added to the unary $\theta^v_p(y^v_p)$ during fusion, since the label y^v_q is known. The parse of the production u with the lowest energy over $\mathcal{R}_X$ is then chosen as its parse (line 13) and the union of the parses of u over $\mathcal{R}_X$ and $\mathcal{R}_{\overline{X}}$ provides the parse of the full image for this region. Convergence would still be guaranteed if INFERSSPN did not parse $\mathcal{R}_{\overline{X}}$ for each production, however, the behavior of the algorithm would be largely degraded as regions would not be able to grow or change in further iterations. At the end of the upward pass, the parse of the image $\hat{t}$ is simply the parse of the start symbol's single region, which contains all pixels (line 14).

3.3. Analysis

As shown in Theorem 1, INFERSSPN always converges to a local minimum of the energy function. Like other move-making algorithms, INFERSSPN explores an exponentially large set of moves at each step, so the returned local minimum is much better than those returned by more local procedures (Boykov et al., 2001), such as max-product belief propagation. Further, we observe convergence in fewer than ten iterations in all experiments, with the majority of the energy improvement occurring in the first iteration.

Theorem 1. *Given a parse (tree) $\hat{t}$ of S over the entire image with energy $E(\hat{t})$, each iteration of* INFERSSPN *constructs a parse (tree) t of S over the entire image with energy $E(t) \leq E(\hat{t})$ and, since the minimum energy of an image parse is finite,* INFERSSPN *will always converge.*

As shown in Proposition 2, each iteration of INFERSSPN has worst-case complexity $O(|G|c(n)n)$, where n is the number of pixels in the image and $c(n)$ is the complexity of the underlying graph cut algorithm used, which is worst-case low-order polynomial, but nearly linear-time in practice (Boykov & Kolmogorov, 2004; Boykov et al., 2001). The additional factor of n is due to the number of regions of each symbol, which in the worst case can be $O(n)$ but in practice is almost always a small constant (often one). Thus, INFERSSPN typically runs in time $O(|G|c(n))$.

Proposition 2. *Let $c(n)$ be the time complexity of computing a graph cut on n pixels and $|G|$ be the size of the grammar defining the SSPN. Then each iteration of* INFERSSPN *takes time $O(|G|c(n)n)$.*

Note that a straightforward application of α-expansion to image parsing that uses one label for every possible parse in the grammar requires an exponential number of labels in general, and thus has exponential time complexity.

INFERSSPN can be extended to productions with more than two constituents by simply replacing the internal graph cut used to fuse subtrees with a multi-label algorithm such as α-expansion. INFERSSPN would still converge because each subtree would still never decrease in energy. An algorithm such as QPBO (Kolmogorov & Rother, 2007) could also be used, which would relax the submodularity requirement.

4. Learning

An SSPN is parameterized by its production costs $\mathbf{w}_\text{s} = \{w_v : v \in R\}$, which are the log-space version of an SPN's weights, and its MRF weights $\mathbf{w}_\text{m}$. The parameters of an SSPN can be learned in a multitude of ways, but we propose here an approach that builds on the insight of Section 3: given the regions of each symbol an SSPN reduces to an SPN, whereas given the production choices an SSPN reduces to an MRF. Similarly, we propose to learn SSPN parameters using an alternating-minimization-style approach like ADMM (Boyd et al., 2011), where the SPN weights $\mathbf{w}_\text{s}$ are first updated with the MRF parameters held fixed and then the MRF weights $\mathbf{w}_\text{m}$ are updated with the SPN weights held fixed, with this process iterating until convergence. While it is possible to learn $\mathbf{w}_\text{s}$ and $\mathbf{w}_\text{m}$ simultaneously, our preliminary investigations indicate

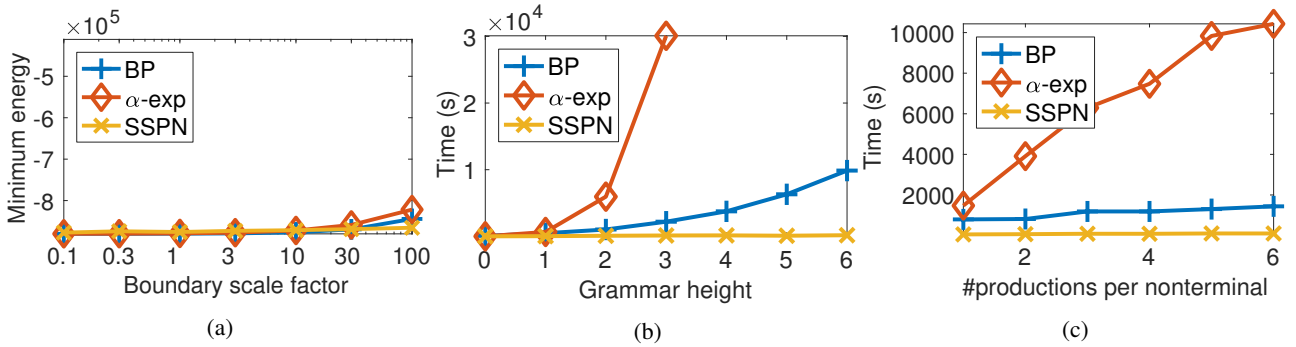


Figure 3. The energy of the returned parse and the total running time when testing inference for each of belief propagation, α-expansion, and INFERSSPN when varying (a) boundary strength, (b) grammar height, and (c) number of productions. Each data point is the average value over (the same) 10 images. Missing data points indicate out of memory errors. Figures 1, 2, and 3 in the supplement show all results for each experiment

that learning them separately provides a more stable approach, where symbols are first associated with different image patches (or features) by updating $\mathbf{w}_{\text{s}}$, and then each symbol's region and appearance weights $\mathbf{w}_{\text{m}}$ are fit to that symbol's image patches (or features).

For weight updating, both SPNs and MRFs can be learned both generatively and discriminatively, and SSPNs are no different; we focus here on the discriminative case, but the generative case is similar. As with both SPNs and MRFs, the derivative of the conditional log-likelihood of an SSPN with respect to a weight w_i is simply the difference of the expected count of the corresponding production (or pixel or edge) over all parse trees that are compatible with both the labels and the image and the expected count over parse trees compatible with only the image; i.e., $\frac{\partial}{\partial w_i} \log p_{\mathbf{w}}(\mathbf{y}|\mathcal{I}) = \mathbb{E}_{t \in \mathcal{T}_{\mathbf{w}}(\mathbf{y},\mathcal{I})}[n_i(t)|\mathbf{y},\mathcal{I}] - \mathbb{E}_{t' \in \mathcal{T}_{\mathbf{w}}(\mathcal{I})}[n_i(t')|\mathcal{I}]$, where $\mathbf{y}$ are the query variables, $\mathcal{T}_{\mathbf{w}}(\mathbf{y},\mathcal{I})$ and $\mathcal{T}_{\mathbf{w}}(\mathcal{I})$ are the sets of parse trees compatible with their respective arguments, and $n_i(t)$ is the count of weight w_i in t. Unfortunately, since no datasets of parsed images exist to train on, both expectations are intractable. However, an effective approximation to the second expectation is to simply use the counts from the MAP parse, which is accurate if it has much of the probability mass; this is known as voted perceptron (Collins, 2002) and has been used to efficiently train both MRFs (Singla & Domingos, 2005) and SPNs (Gens & Domingos, 2012). In SPNs, both expectations are tractable but are still replaced with their MAP state to overcome vanishing gradients. We propose to extend this method to SSPNs, and approximate each expectation with the counts from its respective MAP parse, as found by INFERSSPN. The gradient update is then simply $\frac{\partial}{\partial w_i} \log p_{\mathbf{w}}(\mathbf{y}|\mathcal{I}) \approx n_i(t^*_{\mathbf{y}\mathcal{I}}) - n_i(t^*_{\mathcal{I}})$, where $t^*_{\cdot} = \arg\min_{t \in \mathcal{T}_{\mathbf{w}}(\cdot)} E(t, \mathcal{I})$.

5. Experiments

We evaluated SSPNs by parsing images from the Stanford background dataset (SBD) (Gould et al., 2009) under two different settings, both using features from DeepLab (Chen et al., 2015; 2016), a state-of-the-art convolutional semantic segmentation approach. In the first, we compare the performance of INFERSSPN to α-expansion and belief propagation in an inference-only setting. In the second, we induce grammars and compare the segmentation performance of SSPNs with DeepLab and DeepLab plus a submodular MRF.

Inference evaluation. To evaluate the inference performance of INFERSSPN, we programmatically generated grammars while varying the height, number of productions per nonterminal, and strength of the boundary (pairwise) terms. Note that these grammars are not learned and simply provide a common basis on which to compare inference performance. Each algorithm is given the same grammar and the same features (from DeepLab). Evaluation is performed on the training images in order to separate inference performance from generalization performance. We compared INFERSSPN to α-expansion on a flat pairwise MRF containing a label for each possible parse path in the grammar and to max-product belief propagation (BP) on a multi-level (3-D) pairwise MRF with the same height as each grammar. We used a single weight w_{BF} to parameterize the MRF of every production in conjunction with the standard contrast-dependent boundary term of Shotton et al. (2006). Further details on these models and the MRF parameterization are provided in the supplement. We note that, due to the flat encoding, α-expansion must iterate over an exponential number of labels. However, once it converges, its energy is within a constant factor of the global minimum (Boykov et al., 2001) and thus serves as a good surrogate for the true global minimum, which is intractable to compute.

Increasing the boundary strength of an MRF makes inference more challenging, as individual pixel labels cannot be flipped easily without large side effects. Figure 3a plots the average minimum energy of the parses found by each algorithm versus w_{BF} (with log-scale x-axis). INFERSSPN returns comparable or better parses to both BP and α-expansion and in less time, as shown in Figure 1 in the supplement. Similarly, increasing the height of the grammar can make inference far more challenging, as the number of parse paths in the grammar increases exponentially with height. Here, we set $\mathbf{w}_{\text{BF}}$ to 20 and plot inference time versus grammar height in Figure 3b. As expected from our theoretical results, inference time for INFERSSPN scales linearly with height, whereas it scales exponentially for both α-expansion and BP. Again, the energies and accuracies of the parses returned by INFERSSPN are nearly identical to those of α-expansion, as shown in Figure 2 in the supplement. Finally, the number of paths in the grammar is also directly affected by the number of productions per symbol. We again set w_{BF} to 20 and plot inference time versus number of productions per nonterminal in Figure 3c. Again, INFERSSPN returns equivalent parses to α-expansion and BP in much less time.

Model evaluation. While we do not yet have full SSPN learning in place – mainly due to the difficulty of full-fledged grammar induction and the lack of training data – we are currently working on it. Instead, in order to evaluate the benefits of SSPNs as a model for scene understanding, we induced grammars on the SBD test images and computed the mean pixel accuracy of the terminal labeling (i.e., the per-pixel semantic labels) from the parse tree returned by INFERSSPN. These results and those of DeepLab alone and DeepLab plus a flat (planar) MRF are shown in Table 1, which shows a 20% relative decrease in error for SSPNs, which is quite remarkable given how well the DeepLab features do on their own and how little the flat MRF helps. The goal of this test is not to establish a new state-of-the-art result on this dataset (although all three of these numbers are well-above state of the art), since the SSPNs were induced on subsets of the data, with some oracle information; rather, the goal is to show the promise of SSPNs without requiring full-fledged learning. In particular, to generate each image grammar we first over-segmented each image at 4 different levels of granularity using the method and code of Isola et al. (2014) and intersected the most fine-grained of these with the label regions. For each image, we took its over-segmentations and those for four other randomly chosen images and created a grammar with a symbol for each segment, where each terminal region produced only those labels in its region. Finally, we added productions between overlapping segments at different granularity levels for each image and then randomly added productions between cross-image segments with overlapping regions. On average, each induced grammar had 860 symbols and 1250 productions with 5 constituents. While the induced SSPNs benefit from a small amount of oracle information – in the sense that the induced grammar can only produce terminal labels that are possible at that pixel (much like in the experiments of Russell et al. (2010)) – they also suffer from not being learned, since all production costs were zero and the same MRF parameters were used across all images. Learning these should only further improve performance; in particular, note that the extra per-pixel label information is exactly what would be learned by a good learning algorithm. Thus, we believe that this result demonstrates the promise and power of SSPNs and the benefits of reasoning about higher-level relationships for scene understanding.

Table 1. Mean pixel accuracy on 143 test images of the SBD.

DeepLab	DeepLab+MRF	SSPN(+oracle)
87.46	87.60	**90.03**

6. Conclusion

This paper proposed submodular sum-product networks (SSPNs), a novel extension of sum-product networks that combines their expressivity and power with the efficient combinatorial optimization capabilities of submodular Markov random fields. SSPNs can be understood as an instantiation of an image grammar in which all possible parses of an image over arbitrary shapes are represented. Despite this complexity, we presented INFERSSPN, a move-making algorithm that exploits submodularity in order to find the (approximate) MAP state of an SSPN, which is equivalent to finding the (approximate) optimal parse of an image. Analytically, we showed that INFERSSPN is both very efficient – each iteration takes time linear in the size of the grammar, the image, and the complexity of one graph cut – and convergent. Empirically, we showed that INFERSSPN achieves accuracies and energies comparable to α-expansion, which returns optima within a constant factor of the global optimum, while taking exponentially less time to do so, and that it shows great promise for scene understanding. We have begun work on learning SSPNs from data. This is an exciting avenue of research because many recent works have demonstrated that learning both the structure and parameters of SPNs from data is feasible and effective, despite the well-known difficulty of grammar induction. We also plan to apply SSPNs to activity recognition, social network modeling, and probabilistic knowledge bases.

Acknowledgments

AF would like to thank Robert Gens, Rahul Kidambi, and Gena Barnabee for useful discussions, insights, and as-

sistance with this document. This research was partly funded by ONR grant N00014-16-1-2697 and AFRL contract FA8750-13-2-0019. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ONR, AFRL, or the United States Government.

References

Ahuja, Ravindra K., Magnanti, Thomas L., and Orlin, James B. Network flows: theory, algorithms and applications. *Network*, 1:864, 1993.

Boutilier, Craig, Friedman, Nir, Goldszmidt, Moises, and Koller, Daphne. Context-specific independence in Bayesian networks. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 4, 1996.

Boyd, Stephen, Parikh, Neal, Chu, Eric, Peleato, Borja, and Eckstein, Jonathan. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

Boykov, Yuri and Kolmogorov, Vladimir. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

Boykov, Yuri, Veksler, Olga, and Zabih, Ramin. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

Chandrasekaran, V., Srebro, N., and Harsha, P. Complexity of inference in graphical models. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, pp. 70–78, 2008.

Chen, Liang-Chieh, Papandreou, George, Kokkinos, Iasonas, Murphy, Kevin, and Yuille, Alan L. Semantic image segmentation with deep convolutional nets and fully connected CRFs. *Proceedings of the International Conference on Learning Representations*, 2015. URL http://arxiv.org/abs/1412.7062.

Chen, Liang-Chieh, Papandreou, George, Kokkinos, Iasonas, Murphy, Kevin, and Yuille, Alan L. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. In *ArXiv e-prints*, 2016. URL http://arxiv.org/abs/1412.7062.

Chomsky, Noam. On certain formal properties of grammars. *Information and Control*, 2:137–167, 1959. ISSN 07745141.

Collins, Michael. Discrimative training methods for hidden Markov models: theory and experiments with perceptron algorithms. *Proceedings of the Conference on Empirical Methods in NLP (EMNLP 2002)*, pp. 1–8, 2002.

Dechter, Rina and Mateescu, Robert. AND/OR search spaces for graphical models. *Artificial intelligence*, 171: 73–106, 2007.

Friesen, Abram L. and Domingos, Pedro. The sum-product theorem: A foundation for learning tractable models. In *Proceedings of the 33rd International Conference on Machine Learning (ICML 2016)*, volume 48, 2016.

Gens, Robert and Domingos, Pedro. Discriminative learning of sum-product networks. In *Advances in Neural Information Processing Systems*, pp. 3239–3247, 2012.

Gens, Robert and Domingos, Pedro. Learning the structure of sum-product networks. In *Proceedings of the 30th International Conference on Machine Learning*, pp. 873–880, 2013.

Gould, Stephen, Fulton, Richard, and Koller, Daphne. Decomposing a scene into geometric and semantically consistent regions. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1–8, 2009.

Greig, D. M., Porteous, B.T., and Seheult, A. H. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society. Series B (Methodological)*, 51(2):271–279, 1989.

Hopcroft, John and Ullman, Jeffrey. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading MA, 1979.

Isola, Phillip, Zoran, Daniel, Krishnan, Dilip, and Adelson, Edward H. Crisp boundary detection using pointwise mutual information. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014.

Jurafsky, Daniel S. and Martin, James H. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, 2000.

Kempe, David, Kleinberg, Jon, and Tardos, Éva. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, D.C., 2003. ACM.

Kolmogorov, Vladimir and Rother, Carsten. Minimizing nonsubmodular functions with graph cuts - a review. *IEEE transactions on pattern analysis and machine intelligence*, 29(7):1274–9, 2007.

Kolmogorov, Vladimir and Zabih, Ramin. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26 (2):147–159, 2004.

Komodakis, Nikos, Tziritas, Georgios, and Paragios, Nikos. Fast, approximately optimal solutions for single and dynamic MRFs. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.

Lempitsky, Victor, Rother, Carsten, Roth, Stefan, and Blake, Andrew. Fusion moves for Markov random field optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1392–1405, 2010.

Lempitsky, Victor, Vedaldi, Andrea, and Zisserman, Andrew. A pylon model for semantic segmentation. In *Neural Information Processing Systems*, pp. 1–9, 2011.

Mossel, Elchanan and Roch, Sebastien. On the Submodularity of Influence in Social Networks. In *Symposium on Theory of Computing 2007*, pp. 128–134, 2007.

Poon, Hoifung and Domingos, Pedro. Sum-product networks: A new deep architecture. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, pp. 337–346. AUAI Press, 2011.

Russell, Chris, Ladický, Lubor, Kohli, Pushmeet, and Torr, Philip H.S. Exact and approximate inference in associative hierarchical networks using graph cuts. *The 26th Conference on Uncertainty in Artificial Intelligence*, pp. 1–8, 2010.

Sharma, Abhishek, Tuzel, Oncel, and Liu, Ming-Yu. Recursive context propagation network for semantic scene labeling. In *Advances in Neural Information Processing Systems*, pp. 2447–2455, 2014.

Shotton, Jamie, Winn, John, Rother, Carsten, and Criminisi, Antonio. TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. *Proceedings European Conference on Computer Vision (ECCV)*, 3951(Chapter 1):1–15, 2006.

Singla, Parag and Domingos, Pedro. Discriminative training of Markov logic networks. In *Proceedings of the 20th National Conference on Artificial Intelligence*, pp. 868–873, 2005.

Socher, Richard, Lin, Cliff C., Manning, Chris, and Ng, Andrew Y. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning*, pp. 129–136, 2011.

Zhao, Yibiao and Zhu, Song-Chun. Image parsing via stochastic scene grammar. In *Advances in Neural Information Processing Systems*, pp. 1–9, 2011.

Zhu, Song-Chun and Mumford, David. A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Vision*, 2(4):259–362, 2006.