Structured Attention Networks

Yoon Kim*  Carl Denton*  Luong Hoang  Alexander M. Rush
Harvard University

Summary

- Generalize attention to incorporate latent input structure
- Attention over a combinatorial set (segmentations, parse trees) => exact inference via dynamic programming
- Backpropagate through inference => training remains end-to-end
- Attention layers learn “segmenter/parser as a hidden layer”

Attention Networks

- Attention networks: represent input with a variable length memory bank (instead of a fixed-dimensional vector).
- Requires computing attention distribution (“where”) and annotation function (“what”) given source and query.
- Context vector is defined as the expected annotation.

Structured Attention Networks

- Input structure represented with a graphical model over multiple latent variables z = z₁,..., zₙ.
- p(z | x, q; θ): attention distribution over set of structures.
- Compute attention/context vector using embedded inference.

Segmental Attention: Machine Translation

- Latent variables: z = [z₁,..., zₙ] (binary vectors of length T)
- Attention distribution from a 2-state, linear-chain CRF:
  b unary potentials: θ_b(θ) = x_i, W_q, if k = 1 (otherwise 0)
- Pairwise potentials: binary parameters (i.e., b₁₂, b₂₁, b₁₃, b₂₃, b₁₄, b₂₄, b₃₄)
- Annotation function: f(x, z) = ∑ₙₐ p(z = 1 | x, q, θ, x)
- Context vector: c = E[f(x, z)] = ∑ₙₐ p(z = 1 | x, q, θ, x)

Visualization of the source attention distribution for the simple (top left), sigmoid (top right), and structured (bottom left) attention models over a ground truth sentence on the character-to-word translation task. Bottom right shows the manually-annotated alignments.

Backpropagation through Inference

procedure BACKPROP_FORWARD_BACKWARD(θ)
  θ ← 0
  Forward
  for i = 1, ..., n; do
    α[i, z] ← α[i−1, z] ⊕ β[i, z] ⊕ φ[i, z] ⊕ θ[i, z]
  Backward
  for i = n, ..., 1; do
    β[i, z] ← β[i+1, z] ⊕ τ[i, z] ⊕ θ[i, z]
  Marginals
  for i = 1, ..., n; c ∈ C do
    p(x, z = c) = exp(q[c] ⊗ β[i, c] ⊗ − log Z)

Signed Log-Space Semifield

Simple: p(z = c | x, q) = softmax(θ)
Sigmoid: p(z = 1 | x, q) = sigmoid(θ)
Structured: p(z = 1 | x, q) = FORWARD_BACKWARD(θ)

Bleu  Simple  Sigmoid  Structured
Char → Word  12.9  13.1  14.6  14.6
Word → Word  14.1  13.8  14.3

Conclusions and Future Work

- Structured attention networks generalize simple attention and learn interesting internal representations.
- More experiments/models in the paper (tree transduction, question answering).
- Future work:
  Approximate inference (e.g., loopy belief propagation) for richer graphical models.
  Differentiable optimization as a neural network layer.

Code: https://github.com/harvardnlp/structured-attention

Syntactic Attention: Natural Language Inference

- Input structure: graph-based dependency parser with latent variables zⱼ, ∀i ≠ j (zⱼ = 1 => xᵢ is head of xⱼ)
- Potentials: θⱼ = MLP([hᵢ, hⱼ]), b = BiLSTM(xⱼ)
- Annotation function (parent word): fⱼ(x, z) = ∑ⱼ zⱼ (zⱼ = 1) xⱼ
- Run inside-outside version of Eisner’s algorithm to get marginals:
  p(zⱼ = 1 | x) = INSIDE_OUTSIDE(θ)

- Use parsing marginals to obtain context vector (“soft-parent”):
  cⱼ = E[fⱼ(x, z)] = ∑ⱼ p(zⱼ = 1 | x) xⱼ

Baseline: only word embeddings (i.e. xᵢ)
Hard parent: word + parent from pipelined dependency parser
Syntactic attention: Natural Language Inference

\* Authors contributed equally.