Large-scale deep learning

- Fast CPU Implementations
- GPU Implementations

  Operating System perspective, GPU has lower clock speed and less branching capacity, thus intensive and independent computing can use GPU.

- Large-Scale Distributed Implementations

  Model parallelism or data parallelism

- Model compression
- Dynamic structure
  - The idea behind dynamic structure is that focusing on the part, which influences the results the most (data or model)
  - A cascade of classifiers can efficiently separate the complicated task.
  - Mixture of experts (demonstrated in PRML)
  - Context and attention mechanism: Focus on the weighted part of hidden/input data (same idea with data partition)
- Specialized Implementation

Computer Vision

Most of the computer vision tasks can be seen as expansion of object recognition and object detection.

- Preprocessing
  - Formatting the image to the same scale ⇒ only strictly necessary stuff.
  - Data augmentation without changing or losing key information.

  Two examples of data preprocessing in CV: (Dataset augmentation omitted)

  — Contrast Normalization: Formula reads below

$$X'_{i,j,k} = s \frac{X_{i,j,k} - \bar{X}}{\max\left\{\epsilon, \sqrt{\lambda + \frac{1}{3rc}\sum_i \sum_j \sum_k (X_{i,j,k} - \bar{X})^2}\right\}}$$

scaling

$X_{i,j,k} - \bar{X}$ → normalize the pixels

ensuring the denominator not be zero

λ: shift factor

$\sum_i \sum_j \sum_k (X_{i,j,k} - \bar{X})^2$ → The standard variance of pixels

Contrast normalization is for rescaling all the pixels into the same scale, and the λ, ϵ, s, parameters is for ensuring the computational stability and decreasing the effect of noise.

- Global Contrast Norm vs Local Contrast Norm

Global Contrast Norm: focus more on scaling all the pixels within one region for stabling.

Local Contrast Norm: More sensitive to edge, cause in small windows, we often get local invariant.

Speech Recognition

General method: HMM-GMM → Conv + others → RNN LSTM

Tasks of speech recognition are different from NLP on, it considering converting input radio wave into human understandable format, phonemic or words.

Natural Language Processing (NLP)

Many NLP applications are based on language models that define a probability distribution over sequence of words, characters, or bytes in natural language.

- n-grams

Taking use of chain rule: $P(x_1, \dots x_T) = P(X_1 \sim X_{n-1}) \prod_{t=n}^{T} P(X_t | X_{t-(n-1)}, \dots, X_{t-1})$

Modeling a probability of a whole sequence.

Here we don't consider application specification, we focus on how modeling, a probability of sequence/sentence is calculated by recurrently applying chain rule with every n words.

One way to view n-gram is to regard it as kNN cause the transition probability is actually calculated based on counting on training dataset.

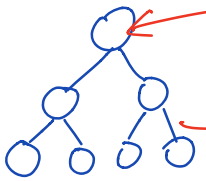
- Neural Language Models

  Embedding word into a reasonable subspace, unlike other tasks, data in NLP often comes discrete and non-digital. (Note that "embedding" is a different task from sequential modeling)

- How we deal with high dimensional output
  - Taking use of a shorted List L with its tail T.

    L (shorter list) reduce the dimension of model and Tail (T) is used as n-gram of extra output (rare vocabularies)
  - Hierarchical softmax

    Using a tree structure to representing vocabularies and when inferencing, output actually searches the best-matched path from root node to the final leaf (words predictions)



  - Importance Sampling

Aimed to use sampling method to reduce the computational cost of $\frac{\partial \log P(y|c)}{\partial \theta}$

$$\frac{\partial \log(P(y|c))}{\partial \theta} = \frac{\partial}{\partial \theta} \log \frac{e^{a_y}}{\sum_i e^{a_i}} = \frac{\partial a_y}{\partial \theta} - \sum_i P(y=i|c) \frac{\partial a_i}{\partial \theta}$$

Here is a form of expectation with can be sampling by weights of importance sampling $\sum_i P(y=i|c) \frac{\partial a_i}{\partial \theta} \doteq \frac{1}{m} \sum_i w_i \frac{\partial a_{n_i}}{\partial \theta_i}$

- Noise-Contrastive Estimation and Ranking Loss
- Combing NLM with n-gram
  - Ensembling : Error independent
  - Adding extra input to output links to calculate n-gram.

- Neural Machine Translation

  Populating encoder-decoder style model

  - Attention mechanism

    $c = \sum_t \alpha_t h_t$ → Can also be input } ⇒ This yields each t has an attention of particular location of h.

    Re-weighting each h_t / input's contribution

    If we select desired h_t without reweighting, the problem is not differentiable.

- Word embedding can be seen as data pre processing for these sequential based model, for better performance.

Other Applications

- Recommending System (Supervised most)
  - collaborative filtering (Bi-linear model or SVD)
  - content-based recommender (For dealing with new-seen data)
  - exploitation (2nd) vs exploration (1st)

    Study within current knowledge → Expanding our knowledge scope
  - Knowledge Representation, reasoning and question answering
    - learning relations ($S=\{(A,B),(C,D),(A,C)\}$, tuples)
    - Training data preparation → relational database

      Model structure → Inspired by NLP