10.1. Attention Mechanism

Lecture based on “Dive into Deep Learning” **http://D2L.AI** (Zhang et al., 2020)

Prof. Dr. Christoph Lippert

Digital Health & Machine Learning

Database queries

Databases are collections of keys (k) and values (v).
$\mathcal{D} = \{$(“Schumacher”, “Anna”), (“Schneider”, “Michael”), (“Firscher”, “Julia”), (“Weber”, “Sarah”), (“Schulz”, “Lisa”), (“Schuler”, “Tim”)$\}$, with the last name being the key and the first name being the value.

- exact query (q) “Schulz” would return the value “Lisa”.
- if (“Schulz”, “Lisa”) not in $\mathcal{D}$, there would be no valid answer.
- For approximate queries, we could retrieve (“Schuler”,“Tim”) instead.

- queries q operate on (k,v) pairs regardless of the database size.
- same query have different answers, according to the contents of the database.
- Simple queries executed on a large state space (the database) (e.g., exact match, approximate match, top-k).

The *attention mechanism* translates this concept to deep learning.

Attention Pooling

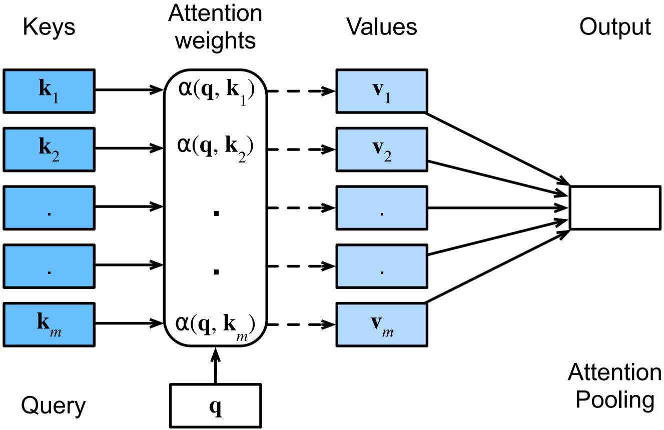
$\mathcal{D} \stackrel{\text{def}}{=} \{(\mathbf{k}_1, \mathbf{v}_1), \ldots (\mathbf{k}_m, \mathbf{v}_m)\}$ a database of m *keys* and *values*.
Let $\mathbf{q}$ be a *query*.
Then we can define the *attention* over $\mathcal{D}$ as

$$\text{Attention}(\mathbf{q}, \mathcal{D}) \stackrel{\text{def}}{=} \sum_{i=1}^{m} \alpha(\mathbf{q}, \mathbf{k}_i)\mathbf{v}_i,$$

where $\alpha(\mathbf{q}, \mathbf{k}_i) \in \mathbb{R}$ $(i = 1, \ldots, m)$ are scalar attention weights.

- *attention* operation pays particular attention to the terms with large weight α.
- Attention over $\mathcal{D}$ generates a linear combination of values in the database.

Attention Pooling



Special Cases

- For nonnegative $\alpha(\mathbf{q}, \mathbf{k}_i)$, the output of the attention mechanism is contained in the convex cone spanned by the values $\mathbf{v}_i$.
- The weights $\alpha(\mathbf{q}, \mathbf{k}_i)$ form a convex combination, i.e., $\sum_i \alpha(\mathbf{q}, \mathbf{k}_i) = 1$ and $\alpha(\mathbf{q}, \mathbf{k}_i) \geq 0$ for all i.
- Exactly one of the weights $\alpha(\mathbf{q}, \mathbf{k}_i)$ is 1, while all others are 0.
- If all weights are equal, i.e., $\alpha(\mathbf{q}, \mathbf{k}_i) = \frac{1}{m}$ for all i, the result is average pooling.

Softmax Attention

Typically, attention weights are normalized to sum up to 1

$$\alpha(\mathbf{q}, \mathbf{k}_i) = \frac{\alpha(\mathbf{q}, \mathbf{k}_i)}{\sum_j \alpha(\mathbf{q}, \mathbf{k}_j)}.$$

Any function $a(\mathbf{q}, \mathbf{k})$ can be combined with softmax

$$\alpha(\mathbf{q}, \mathbf{k}_i) = \frac{\exp(a(\mathbf{q}, \mathbf{k}_i))}{\sum_j \exp(a(\mathbf{q}, \mathbf{k}_j))}.$$

Benefits of **softmax attention**:

- available in all deep learning frameworks.
- differentiable
- gradient never vanishes

- Alternatives possible
- non-differentiable attention can be trained using reinforcement learning.
- One of the benefits of the attention mechanism is that it can be quite intuitive, particularly when the weights are nonnegative and sum to 1.
- In this case we might *interpret* large weights as a way for the model to select components of relevance.
- While this is a good intuition, it is important to remember that it is just that, an *intuition*.

Summary

- Attention allows us to aggregate data from many (key, value) pairs.
- Attention is a *differentiable* way by which a neural network can select elements from a set and to construct a weighted sum over representations.

Where do queries, keys, and values arise from in Machine Learning?

- in regression (Nadaraya-Watson estimator)
    - the query might correspond to the location where the regression should be carried out.
    - The values are the (regression) values themselves.
- Any of queries, keys and values are differentiable and can be learned from data in different Deep Learning architectues