SMAI Project

Project : From Softmax to Sparsemax

Members : Akhilesh Soni(201530001)
Nikhil Parasaram(201501202)
Raghav Gupta(201501192)
Satyam Mittal(201501020)

Project Description

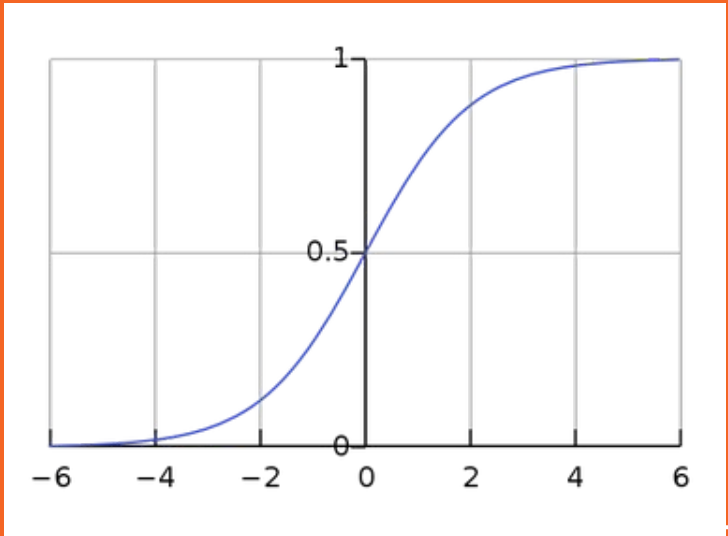
- The paper focuses on sparsemax, a new activation function which is similar to softmax, but outputs sparse probabilities.
- It has properties similar to the softmax and it's Jacobian can be efficiently computed, enabling its use in a neural network trained with backpropagation.
- Then, a new smooth and convex loss function which is the analogue of the logistic loss is defined for sparsemax.
- Promising empirical results are obtained in multi-label classification problems and in attention-based neural networks for natural language inference but with a selective, more compact, attention focus.

Activation Functions

- In neural networks, activation function of a node defines the output of that node given the input values
- They are used in unit of neural networks which enables us to capture interesting properties of input
- They are chosen to be non-linear to model the non-linear classification
- The output layer has a very specific objective – to try to replicate the true labels as much as possible.
- They are numerous functions like sigmoid, tanh, ReLU, softmax, sparsemax, etc

Literature Survey

a) **Sigmoid:**



Formula: $\sigma(x) = 1 / (1 + e^{-x})$

Pros:

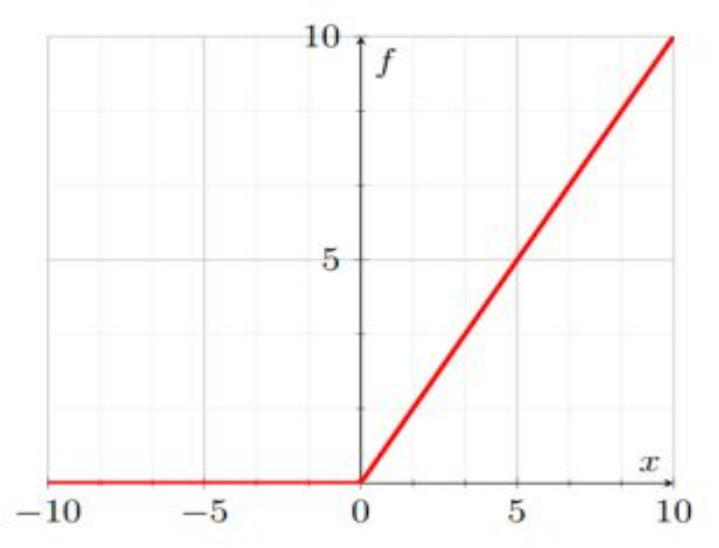
a) Range is between 0 and 1
b) Easy to understand and apply

Cons:

a) Vanishing gradient problem
b) Slow convergence

Literature Survey

b) Relu



Formula: f(x) = max(0, x)

Pros:

a) Accelerates the convergence of SGD compared to sigmoid and tanh
b) No expensive operations are required

Cons:

a) ReLu could result in Dead Neurons

Literature Survey

c) Power Linear Units (PoLUs)

https://arxiv.org/pdf/1802.00212.pdf
February 2018

$$f_n(x) = \begin{cases} x & if \quad x \geq 0 \\ (1-x)^{-n} - 1 & if \quad x < 0 \end{cases}$$

a) For positive values, It is same as ReLU.
b) Removes bias shift effect: Output mean close to 0
c) Saturation in negative part make it noise robust.

d) Swish or Sigmoid-weighted linear unit (SiLU):

$f(x) = x * \sigma(x)$

Softmax

Softmax Activation Function converts a vector of real weights to a probability distribution. It is useful in multi-class classification because it returns the probability vector indicating the probability of example being in the class.

$$Softmax_i(z) = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

Side effects of softmax

- Finding exp(x) is computationally expensive for large values of x therefore computing the value of softmax activation is computationally expensive
- Softmax never (theoretically) outputs zero probability so it does not give any sparse posterior distribution
- For multi label classification softmax does not give good results
- Softmax cannot be used for regression tasks

Sparsemax

Sparsemax Activation Function also converts a vector of real weights to a probability distribution but it outputs sparse distribution i.e it truncates small probabilities to zero. This is very useful in multi-class classification because it returns less number of classes indicating the main classes. It is also useful in attention based neural networks where focus is on the limited set of data instead of the whole.

$$\text{sparsemax}_i(z) = [z_i - \tau(z)]_+$$

Where τ : R^K**->R is the threshold function .**

Sparsemax Algorithm

Input: z

Sort z as $z_{(1)} \geq \ldots \geq z_{(K)}$

Find $k(z) := \max \left\{ k \in [K] \mid 1 + k z_{(k)} > \sum_{j \leq k} z_{(j)} \right\}$

Define $\tau(z) = \frac{\left(\sum_{j \leq k(z)} z_{(j)}\right) - 1}{k(z)}$

Output: p s.t. $p_i = [z_i - \tau(z)]_+$.

Properties

Sparsemax Activation Function retains the desirable properties of the softmax activation function.

- $\rho(0) = 1/K$ and $\lim(\epsilon->0^{+})\rho(\epsilon^{-1}z) = A(z)/|A(z)|$. For sparsemax, the last equality holds for any $\epsilon \leq \gamma(z) \cdot |A(z)|$.
- $\rho(z) = \rho(z + c1)$, for any $c \in R$
- $\rho(Pz) = P\rho(z)$ for any permutation matrix P
- If $zi \leq zj$, then $0 \leq \rho j\,(z) - \rho i\,(z) \leq \eta(zj - zi\,)$, where $\eta = 1/2$ for softmax, and $\eta = 1$ for sparsemax.

Advantages of Sparsemax over Softmax

- Returns sparse posterior distributions i.e assigns exactly zero probability to some of the output variables.
- Jacobian of sparsemax is easy to compute and leads to faster gradient backpropagation.
- Sparsemax loss is convex, everywhere differentiable
- Better results on standard datasets in multi-label classification and attention based neural networks in comparison to softmax.

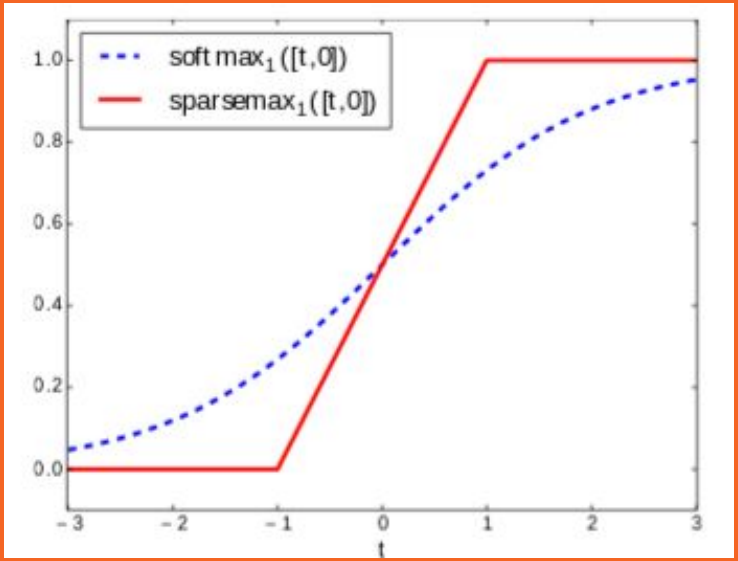
2-D Case

- Consider the input vector as z = (t, 0)
- Then the softmax1(z) reduces to nothing but the sigmoid i.e softmax(z) = $\sigma(t) = 1 / (1 + e^{-t})$
- In the sparsemax case the output is

$$\text{sparsemax}_1(z) = \begin{cases} 1, & \text{if } t > 1 \\ (t+1)/2, & \text{if } -1 \leq t \leq 1 \\ 0, & \text{if } t < -1. \end{cases}$$

Sparsemax vs Softmax

Comparison of Softmax vs Sparsemax for 2-D Case.



Jacobian Softmax

- **The Jacobian matrix of a transformation ρ, is of key importance to train models with gradient-based optimization**
- **Jacobian of softmax is**

$$\begin{aligned}\frac{\partial \text{softmax}_i(\boldsymbol{z})}{\partial z_j} &= \frac{\delta_{ij} e^{z_i} \sum_k e^{z_k} - e^{z_i} e^{z_j}}{\left(\sum_k e^{z_k}\right)^2} \\ &= \text{softmax}_i(\boldsymbol{z})(\delta_{ij} - \text{softmax}_j(\boldsymbol{z}))\end{aligned}$$

Jacobian Sparsemax

- **The Jacobian matrix of a transformation ρ, is of key importance to train models with gradient-based optimization**
- **Jacobian of sparsemax is :**

$$\frac{\partial \text{sparsemax}_i(z)}{\partial z_j} = \begin{cases} \delta_{ij} - \frac{1}{|S(z)|}, & \text{if } i, j \in S(z) \\ 0, & \text{otherwise.} \end{cases}$$

Convergence

- Jacobian of sparsemax is same as the Laplacian of a graph whose elements of S(z) are fully connected. To compute it, we only need S(z), which can be obtained in O(K) time with the same algorithm that evaluates the sparsemax
- Jacobian of softmax takes O(n) time to compute where n is the number of labels
- Therefore sparsemax leads to faster gradient backpropagation than the softmax, particularly in the case when number of important classes << number of classes

Sparsemax Loss Function

- Sparsemax loss is defined as a sparse analogue of logistic loss function. The definition is

$$L_{\text{sparsemax}}(\boldsymbol{z}; k) = -z_k + \frac{1}{2} \sum_{j \in S(\boldsymbol{z})} (z_j^2 - \tau^2(\boldsymbol{z})) + \frac{1}{2}$$

- The loss function is convex and always greater than or equal to zero
- The loss function is differentiable everywhere and its derivative is given by

$$\nabla_{\boldsymbol{z}} L_{\text{sparsemax}}(\boldsymbol{z}; k) = -\boldsymbol{\delta}_k + \text{sparsemax}(\boldsymbol{z})$$

MultiLabel Classification

- Till now we have discussed about multiclass classification, which has only 1 target label, whereas in multilabel classification there can be multiple labels as target.
- We can consider more general problem of sparse label proportion estimation. The loss here is defined as KL divergence between target labels and activation of the output of network.

$$L_{\text{softmax}}(z; q) = \text{KL}(q \parallel \text{softmax}(z))$$

- Where KL divergence is

$$D_{\text{KL}}(P\|Q) = -\sum_i P(i) \log \frac{Q(i)}{P(i)}$$

MultiLabel Classification

- After expanding it we get a generalization of multiclass classification.

$$\begin{aligned} L_{\mathrm{softmax}}(\boldsymbol{z}; \boldsymbol{q}) &= \mathbf{KL}(\boldsymbol{q} \parallel \mathrm{softmax}(\boldsymbol{z})) \\ &= -\mathbf{H}(\boldsymbol{q}) - \boldsymbol{q}^\top \boldsymbol{z} + \log \textstyle\sum_j \exp(z_j), \end{aligned} \tag{24}$$

- After taking the gradient for the loss the H(q) term(shannon entropy) will be eliminated which leads to the above equation which is generalized form of the equation for multiclass classification.

$$\nabla_{\boldsymbol{z}} L_{\mathrm{softmax}}(\boldsymbol{z}; \boldsymbol{q}) = -\boldsymbol{q} + \mathrm{softmax}(\boldsymbol{z}).$$

MultiLabel Classification

- We will generalize the same result for sparsemax as the following.

$$L_{\text{sparsemax}}(\boldsymbol{z};\boldsymbol{q}) = -\boldsymbol{q}^\top \boldsymbol{z} + \frac{1}{2}\sum_{j\in S(\boldsymbol{z})}(z_j^2 - \tau^2(\boldsymbol{z})) + \frac{1}{2}\|\boldsymbol{q}\|^2,$$

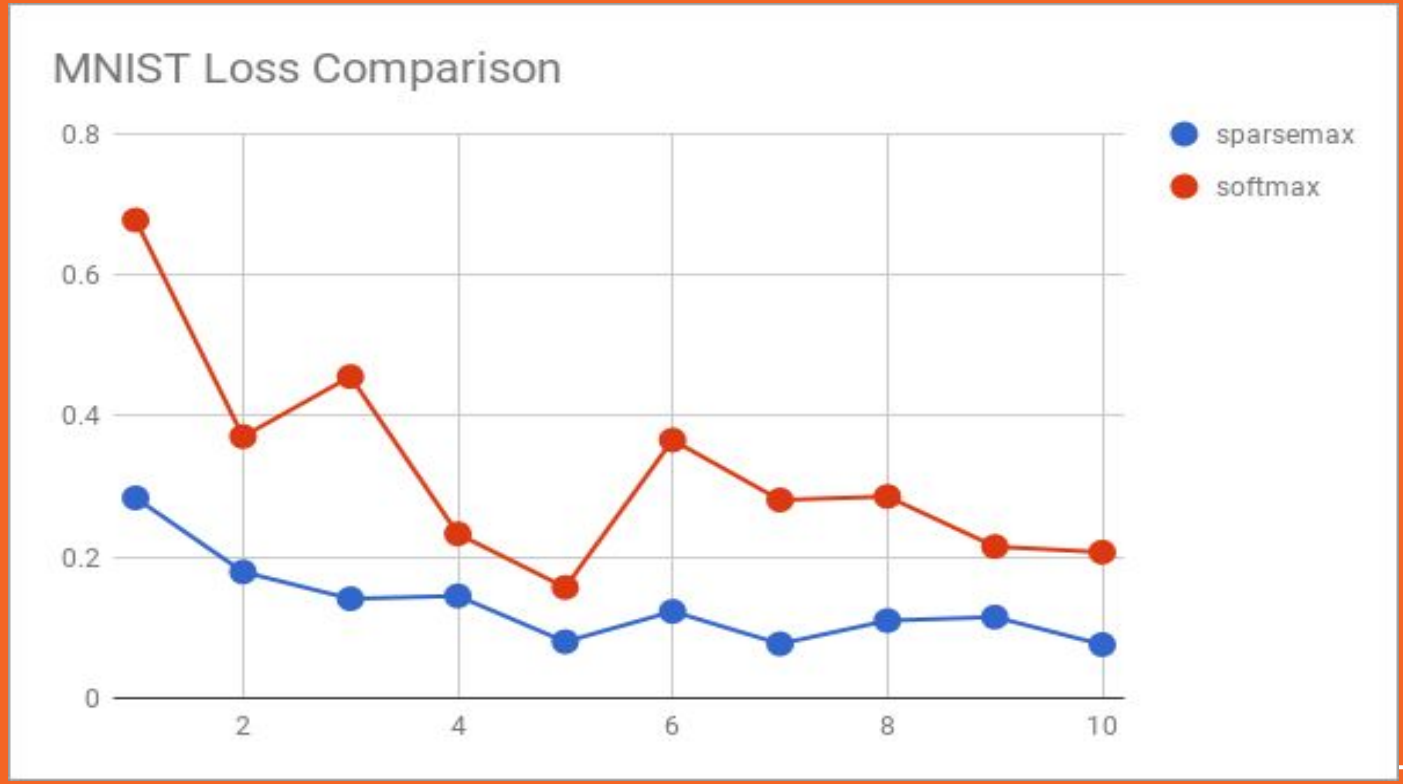
- After taking the gradient we will end up with the generalized expression of multiclass loss gradient , which is quite interesting.

$$\nabla_{\boldsymbol{z}} L_{\text{softmax}}(\boldsymbol{z};\boldsymbol{q}) = -\boldsymbol{q} + \text{softmax}(\boldsymbol{z}).$$

Experiments

Dataset	Softmax	Sparsemax	Logistic
Scene(6 labels)	**72.8%**	72.6%	68.4%
Emotions (6 labels)	64.3%	64.1%	**64.5%**
Birds (19 labels)	44.1%	**44.3%**	41.7%
CAL500(174 labels)	41.3%	**42.5%**	43.1%
Reuters(100 labels)	75.6%	**76.2%**	75.1%

MNIST Dataset (loss vs epochs)



Attention based Learning

- These techniques are mainly used in text related domains(can also be used for images).
- In this method the model trains itself to find important words in the sentences, rather than considering the entire sentence.
- With this it will be helpful in lot of tasks like neural machine translation, NL inference problem, generating image descriptions (link) .
- Sparsemax will be better than softmax as an activation in this case because of its ability to generate sparse probabilities only some words(or parts in an image) are selected which would be the key for making the final decision.

NL inference problem

- We will be given a premise and a hypothesis.
- Using these we need to say whether the statements entail or contradict each other or are neutral.

A boy **rides on** a **camel** in a crowded area while talking on his cellphone.
Hypothesis: *A boy is riding an animal.* [entailment]

A young girl wearing **a pink coat** plays with a **yellow** toy golf club.
Hypothesis: *A girl is wearing a blue jacket.* [contradiction]

Two black dogs are **frolicking** around the **grass together**.
Hypothesis: *Two dogs swim in the lake.* [contradiction]

A man wearing a yellow striped shirt **laughs** while **seated next** to another **man** who is wearing a light blue shirt and **clasping** his hands together.
Hypothesis: *Two mimes sit in complete silence.* [contradiction]

- The above highlighted words are the ones to which sparsemax gave attention to.

NL inference experiment

Model	Validation	Test
No Attention	80.42%	78.3%
Logistic Attention	80.28%	78.56%
Softmax Attention	81.4%	80.9%
Sparsemax Attention	81.27%	81.02%

Conclusion

- Introduced sparsemax transformation having similar properties as softmax but able to output sparse probabilities
- Derived its jacobian needed for the backpropagation algorithm
- Defined a sparsemax loss function which is convex and differentiable
- Empirical results shown in multi-label classification, and NL inference.

References

- Research paper: From softmax to sparsemax https://arxiv.org/pdf/1602.02068.pdf
- Statistical Learning with Sparsity : the Lasso and Generalisations.
- Optimisation and NonSmooth Analysis by Clark Frank.
- Pattern Classification by Richard O. Duda.
- Pattern Recognition and machine learning by Christopher Bishop

Thank You !