1.1 What is a gate in LSTM

- [x] Sigmoid multiplication to real value
- [ ] Sigmoid multiplication to positive real value
- [ ] ReLU multiplication to real value
- [ ] ReLU multiplication to positive real value

1.2 How many weights does an nn.Conv2D(3,20,5) layer have?

- [ ] 60
- [ ] 300
- [ ] 900
- [x] 1500

1.3 Using nn.averagepool(2, stride=4) on a 200×200 input results in

- [ ] 0% inputs not used
- [ ] 25% inputs not used
- [ ] 50% inputs not used
- [x] 75% inputs not used

Regression Losses

Systematic comparison

	optimizable	outlier-robust	ϵ-tolerant
$0/1$ loss	✗	✓	✓
squared loss $(y-t)^2$	✓	✗	✓
absolute loss $\lvert y-t\rvert$	✓	✓	✗
log-cosh loss	✓	✓	✓

Note:

- Many further loss functions have been proposed in the literature (e.g. Huber's loss, ϵ-sensitive loss, etc.). They often implement similar desirable properties as the log-cosh loss.

1.6 The 0/1 loss $1_{\{yt>0\}}$ is

- [x] Not hard to optimize
- [ ] Not outlier robust
- [ ] ... to small variations
- [ ]

1.7 The perceptron loss can be written as

- [x] $\max(0,-yt)$
- [ ] $\max(0,-yt)^2$
- [ ] $1_{\{yt>0\}}$
- [ ] $1_{\{yt>\varepsilon\}}$ for some $\varepsilon>0$

Systematic comparison

	optimizable	mislabeling-robust	builds margin
$0/1$ loss	✗	✓	✗
perceptron loss $\max(0,-yt)$	✓	✓	✗
log loss $\log(1+\exp(-yt))$	✓	✓	✓

1.8 Small local invariances can be addressed by using

- [ ] Activation layers
- [x] Pooling layers
- [ ] Convolution layers
- [ ] Linear layers

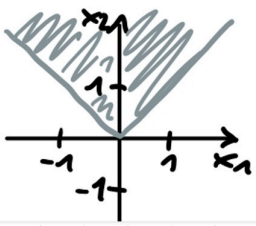
2. Task: Decision boundary

Given: $a_3 = \max(x_1 + x_2)$
$a_4 = \max(x_1 - x_2)$
$y = a_3 + a_4$

a) Draw the four pieces on which y is linear

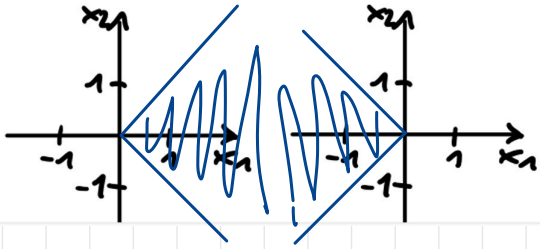
Piece 1 (given)

$(x_2 > -x_1) \wedge (x_2 > x_1)$



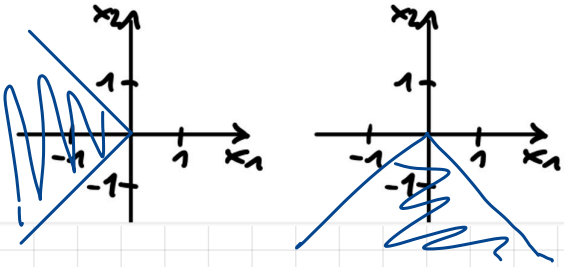
Piece 2

$(x_2 > -x_1) \wedge (x_2 < x_1)$



Piece 3

$(x_2 < -x_1) \wedge (x_2 > x_1)$



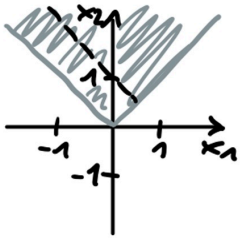
Piece 4

$(x_2 < -x_1) \wedge (x_2 < x_1)$

b) Draw the decision boundary on each piece for $y = f(x) = 1$

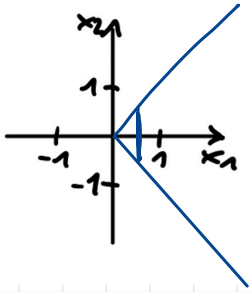
Piece 1 (given)

$y = x_1 + x_2$



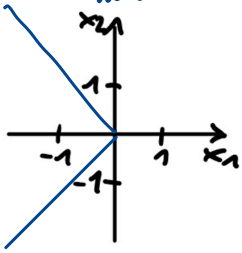
Piece 2

$y = 2x_1$



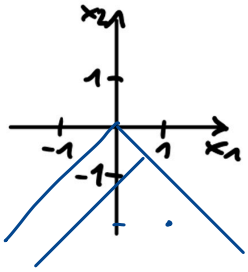
Piece 3

$y = 0 \neq 1$
no solution



Piece 4

$y = x_1 - x_2 = 1$



3. Task: Gradient, bound and regularization

Given: $z_{ij} = |x_i - c_{ij}|$, $a_j = \exp(-\sum_{i=1}^{d} z_{ij})$, $y = \sum_{j=1}^{h} w_j a_j$

a) Calculate $\frac{\partial y}{\partial x_i}$

$$\frac{\partial y}{\partial x_i} = \sum_{j=1}^{h} \frac{\partial y}{\partial a_j} \frac{\partial a_j}{\partial z_{ij}} \frac{\partial z_{ij}}{\partial x_i}$$

$$= \sum_{j=1}^{h} w_j \cdot \exp\left(-\sum_{i=1}^{d} z_{ij}\right) \cdot (-1) \cdot \operatorname{sign}(x_i - c_{ij})$$

$$= -\sum_{j=1}^{h} w_j a_j \operatorname{sign}(x_i - c_{ij})$$

b) Show that $\left\|\frac{\partial y}{\partial x}\right\| \leq d \|w\|_1$ Hint: $|\sum a_i| \leq \sum |a_i|$

$$\left\| \frac{\partial y}{\partial x} \right\| = \sum_{i=1}^{d} \left| \frac{\partial y}{\partial x_i} \right|$$

$$= \sum_{i=1}^{d} \left| \sum_{j=1}^{h} w_j a_j \right|$$

$$\leq \sum_{i=1}^{d} \sum_{j=1}^{h} |w_j| \quad \text{da } a_j \in 1$$

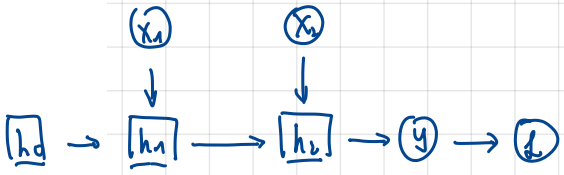
$$= d \, \|w\|$$

C) Explain in 1-2 sentences how this bound helps with choosing the regularization

If we regulate (penalize) $\|w\|$ to be keep it small, then the Gradient $\frac{\partial y}{\partial x}$ is also small which helps avoid overfitting

4. Task: RNN

Given: $h_1 = \tanh(x_1^T w + h_0)$
$h_2 = \tanh(x_2^T w + h_1)$
$y = h_2$
$\mathcal{E} = (y-t)^2$



a) Draw the associated NN graph

b) Calculate $\frac{\partial \mathcal{E}}{\partial y}$

$$\frac{\partial \mathcal{E}}{\partial y} = 2(y-t)$$

c) State the derivative of $\frac{\partial \mathcal{E}}{\partial w}$ using the chain rule

$$\frac{\partial \mathcal{E}}{\partial w} = \frac{\partial \mathcal{E}}{\partial y} \frac{\partial y^t}{\partial h_2} \left(\frac{\partial h_2^t}{\partial w} + \frac{\partial h_2^t}{\partial h_1^t} \frac{\partial h_1^t}{\partial w} \right)$$

d) Calculate $\frac{\partial \mathcal{E}}{\partial w}$

$$= 2(y-t) \cdot 1 \cdot \left(\tanh'(x_2^T w + h_1) \cdot x_2 + \tanh'(x_2^T w + h_1) \cdot \tanh'(x_1^T w + h_0) \, x_1 \right)$$

$$= 2(y-t) \tanh'(x_2^T w + h_1) \left(x_2 + \tanh'(x_1^T w + h_0) x_1 \right)$$

5. Task: Programming

a) Program an adversarial attack on a trained model using $\min_z \{\|x-z\|^2 + \max(0, t \cdot f_{cz})\}$

$x: 1 \times d$
$t: 1 \times d$
$\text{model}(x): 1 \times 1$

use 1000 iterations, stopping not needed

```
1  import torch, import...
2  def adversarial(X, T, model)
3      #Create/modify z
4      z = X.clone().detach().requires_grad_(True)
5
6      optim = torch.nn.SGD([z], 0)
7      #create optimization
8      for i in range(1000):
9          optim.zero_grad()
10         loss = torch.norm(X - z, p=2) ** 2 + torch.max(torch.zero_like(T), T * model(z)).sum()
11
12         loss.backward()
13         optim.step()
14
15     return z
```

(b) $\|x-z\|^2$

(a) $(x-z).\text{pow}(2).\text{sum}()$

b) How would this code need to be modified to take $x \in \mathbb{R}^{mod}$ as input? Use extra logs and provide line numbers!

c) How can the model be made more robust?