https://github.com/sungbinlim/mge51101-instruct/blob/master/coursework/Lecture_5_Deep_Learning_Computation.ipynb

Lesson 5: Deep Learning Computation with PyTorch in a Nutshell

Introduction to Deep Learning

Sungbin Lim (SME)

Assignments

- Revise your proposal in detail (until 4/9) → Determine topics (until 4/23)
  - Explain your Data Science problems in detail
    - What is your goal? Why this is important problem?
    - How to evaluate your model empirically? What is your metric?
  - Describe [what is, how to gather, difficulty of] your data concretely
  - Read the issues in your GitHub and revise your proposal

begin your project

begin your project and complement proposal

complement proposal ***before*** begin your project,

Assignments

you need to explain these things

- Revise your proposal in detail (until 4/9) → D 4/23)
  - Explain your Data Science problems in detail
    - What is your goal? Why this is important pr
    - How to evaluate your model empirically? W
  - Describe [what is, how to gather, difficulty of]
  - Read the issues in your GitHub and revise your proposal

Pre-review

- [x] Convincing data science problems
- [x] Unstructured & large data
- [ ] Description of data
- [x] Difficulty of data
- [ ] Evaluation
- [ ] Data preparation
- [ ] Feasibility

Labels

Approved

begin your project

Labels

Approved

documentation

begin your project and complement proposal

Labels

documentation

question

complement proposal ***before*** begin your project,

Tips

- Focus on few topics
- Don’t suggest vague or mixed goals
- Model and approach is not important in proposal
  - explain your machine learning problem and data in detail
  - prepare your data very well (no synthetic or artificial data)
  - establish quantitative evaluation method clearly
- Write your proposal formally with technical details as possible

Applying the advanced models does not mean better grade in this course

Assignments

- Revise your proposal in detail (until 4/9) → Determine topics (until 4/23)
  - Explain your Data Science problems in detail
    - What is your goal? Why this is important problem?
    - How to evaluate your model empirically? What is your metric?
  - Describe [what is, how to gather, difficulty of] your data concretely
  - Read the issues in your GitHub and revise your proposal

begin your project

begin your project and complement proposal

complement proposal ***before*** begin your project,

Assignments

- Read papers
  - LeNet (1998): Gradient-Based Learning Applied to Document Recognition
  - AlexNet (2012): ImageNet Classification with Deep Convolutional Neural Networks

  until 4/9

  - ZFNet (2013): Visualizing and Understanding Convolutional Networks
  - VGG (2014): Very Deep Convolutional Networks for Large-Scale Image Recognition
  - Inception (2014): Going Deeper with Convolutions
  - ResNet (2015): Deep Residual Learning for Image Recognition
  - WideResNet (2016): Wide Residual Networks
  - DenseNet (2016): Densely Connected Convolutional Networks

  until 4/14

  - NAS (2018): Learning Transferable Architectures for Scalable Image Recognition
  - EfficientNet (2019): Rethinking Model Scaling for Convolutional Neural Networks

  until 4/16

PyTorch Tutorials

- Custom Dataset how can I *prepare* my data?
- Data Preprocessing how can I *preprocess* my data?
- Custom Functions and AutoGrad how can I *implement* custom functions?
- Layers, Modules, and Sequential Blocks how can I *build* my model efficiently?
- PyTorch and multi-GPU how can I *utilize* multi-GPUs?
- Checkpoints how can I *manage* my models?
- Visualization how can I *visualize* loss functions and the results?

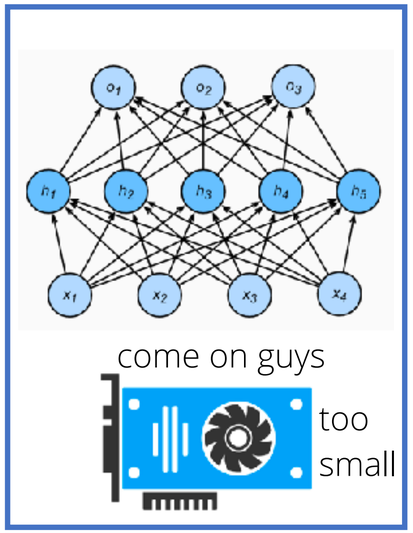
PyTorch Tutorials

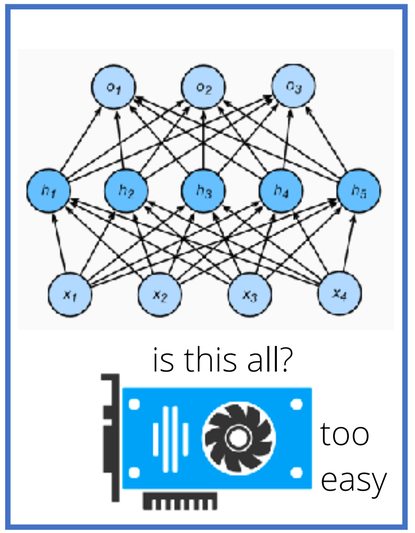
- Custom Dataset how can I *prepare* my data?
- Data Preprocessing how can I *preprocess* my data?
- Custom Functions and AutoGrad how can I *implement* custom functions?
- Layers, Modules, and Sequential Blocks how can I *build* my model efficiently?
- PyTorch and multi-GPU how can I *utilize* multi-GPUs?
- Checkpoints how can I *manage* my models?
- Visualization how can I *visualize* loss functions and the results?

Deep Learning with ~~B~~g Data

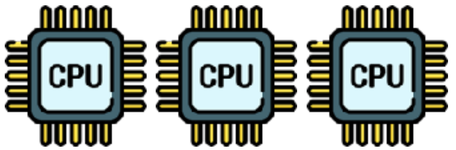
	A	B	C	D	E	F
1	Country	Salesperson	Order Date	OrderID	Units	Order Amount
2	USA	Fuller	1/01/2011	10392	13	1,440.00
3	UK	Gloucester	2/01/2011	10397	17	716.72
4	UK	Bromley	2/01/2011	10771	18	344.00
5	USA	Finchley	3/01/2011	10393	16	2,556.95
6	USA	Finchley	3/01/2011	10394	10	442.00
7	UK	Gillingham	3/01/2011	10395	9	2,122.92
8	USA	Finchley	6/01/2011	10396	7	1,903.80
9	USA	Callahan	8/01/2011	10399	17	1,765.60
10	USA	Fuller	8/01/2011	10404	7	1,591.25
11	USA	Fuller	9/01/2011	10398	11	2,505.60
12	USA	Coghill	9/01/2011	10403	18	855.01
13	USA	Finchley	10/01/2011	10401	7	3,868.60
14	USA	Callahan	10/01/2011	10402	11	2,713.50
15	UK	Rayleigh	13/01/2011	10406	15	1,830.78
16	USA	Callahan	14/01/2011	10408	10	1,622.40
17	USA	Farnham	14/01/2011	10409	19	319.20
18	USA	Farnham	15/01/2011	10410	16	802.00

Tabular data





I can do better than them



Training
- matrix calculus
- backprop
- parallel computing
- ...

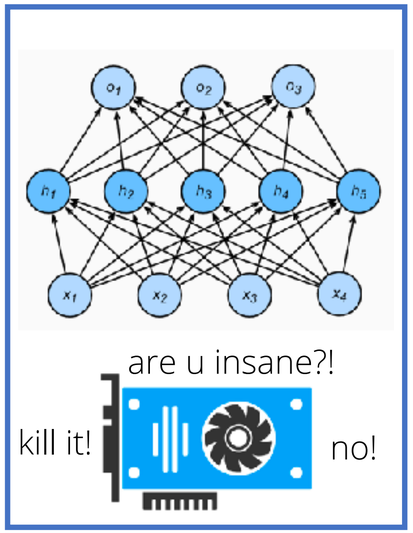
Working
- read & load data
- preprocessing
- aggregate
- ...

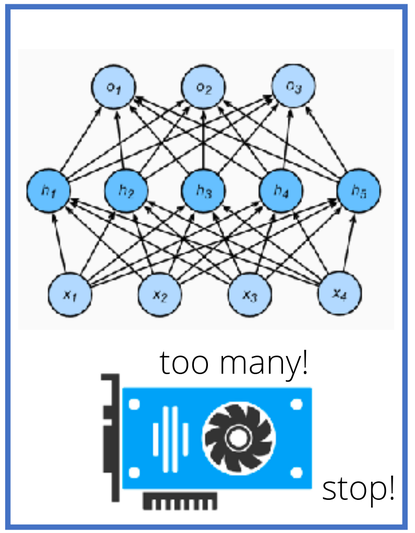
Did I say DL is not good for small data?

Deep Learning with Big Data

$$256 \times 256 \times 3 \times 1{,}000{,}000 \approx 2^{37}$$

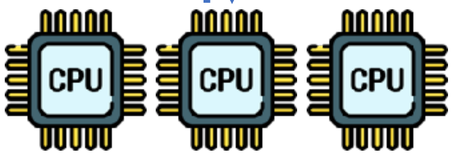
Image data





Training
- matrix calculus
- backprop
- parallel computing
- ...

damn stupid GPUs

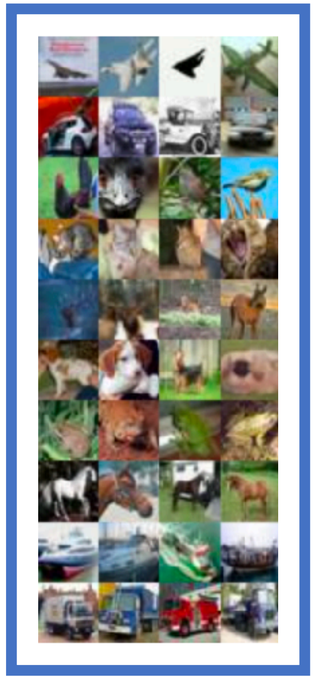


Working
- read & load data
- preprocessing
- aggregate
- ...

If we upload such data at once, then OOM occurs

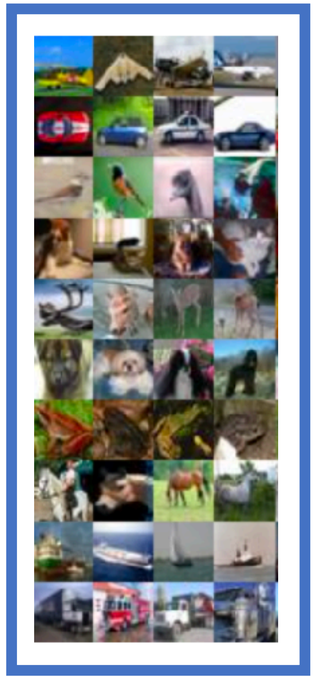
Deep Learning with Big Data

sleeping..



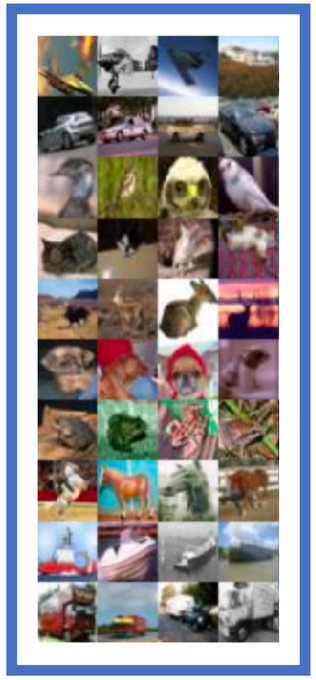
$\mathcal{B}$

get ready!



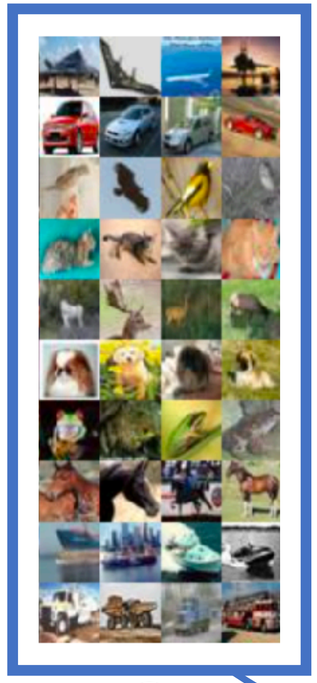
$\mathcal{B}$

I am ready!



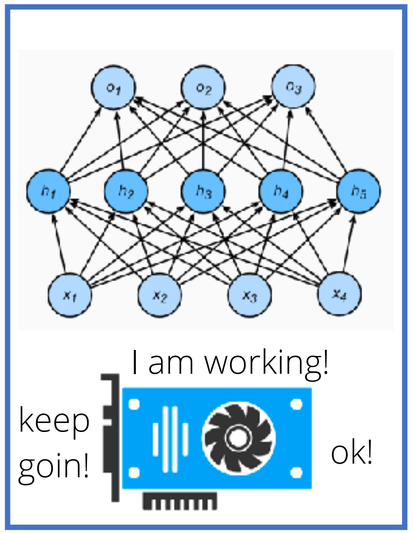
$\mathcal{B}$

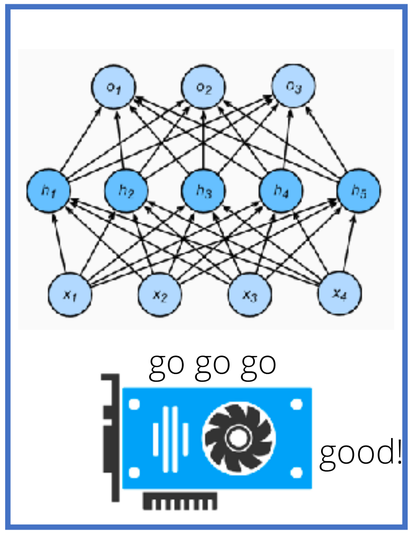
I am going!



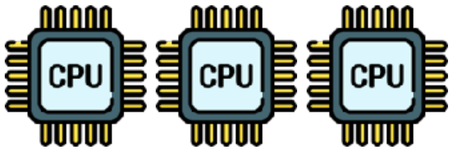
$\mathcal{B}$

$$256 \times 256 \times 3 \times |\mathcal{B}| \leq 2^{18} \cdot |\mathcal{B}| \leq 2^{10}$$





much easier than before



Training
- matrix calculus
- backprop
- parallel computing
- ...

Working
- read & load data
- preprocessing
- aggregate
- ...

we implement this by `generator`

Custom Dataset

- How can I *prepare* my data?
  - How to create a `Dataset` ?

```python
from torch.utils.data import Dataset, DataLoader

class CustomDataset(Dataset):
    """
    Skeleton of Custom Dataset
    Custom Dataset 의 기초 골격
    """

    def __init__(self):
        """
        Initial setting: reading a csv, assigning transforms etc.
        데이터셋 초기화. csv 파일 읽는 부분 또는 전처리를 담당
        """

    def __len__(self):
        """
        Returns the size of the dataset
        데이터셋의 길이
        """

    def __getitem__ (self, idx):
        """
        Extract one sample from a dataset.
        데이터셋에서 1개의 sample 을 추출.
        """
```

Custom Dataset + DataLoader

- How can I *prepare* my data?
  - How to create a `Dataset` ?
  - How to use `DataLoader` ?

```
DataLoader(dataset, batch_size=1, shuffle=False, sampler=None,
           batch_sampler=None, num_workers=0, collate_fn=None,
           pin_memory=False, drop_last=False, timeout=0,
           worker_init_fn=None)
```

use `num_workers` for ***multiprocessing***

```python
from torch.utils.data import Dataset, DataLoader

class CustomDataset(Dataset):
    """
    Skeleton of Custom Dataset
    Custom Dataset 의 기초 골격
    """

    def __init__(self):
        """
        Initial setting: reading a csv, assigning transforms etc.
        데이터셋 초기화. csv 파일 읽는 부분 또는 전처리를 담당
        """

    def __len__(self):
        """
        Returns the size of the dataset
        데이터셋의 길이
        """

    def __getitem__ (self, idx):
        """
        Extract one sample from a dataset.
        데이터셋에서 1개의 sample 을 추출.
        """
```

Data Preprocessing

- Define operations on Dataset
  - call the operations in `__getitem__`

```python
class CustomDataset(Dataset):
    """
    Skeleton of Custom Dataset
    Custom Dataset 의 기초 골격
    """

    def __init__(self):
        """
        Initial setting: reading a csv, assigning transforms etc.
        데이터셋 초기화. csv 파일 읽는 부분 또는 전처리를 담당
        """

    def __len__(self):
        """
        Returns the size of the dataset
        데이터셋의 길이
        """

    def __getitem__ (self, idx):
        """
        Extract one sample from a dataset.
        데이터셋에서 1개의 sample 을 추출.
        """

    def operation(self, *args):
        """
        Implement preprocessing
        """
```

Data Preprocessing

- Define operations on Dataset
  - call the operations in `__getitem__`

```python
class Dataset(data.Dataset):
    """Custom data.Dataset compatible with data.DataLoader."""
    def __init__(self, src_path, trg_path, src_word2id, trg_word2id):
        """Reads source and target sequences from txt files."""
        self.src_seqs = open(src_path).readlines()
        self.trg_seqs = open(trg_path).readlines()
        self.num_total_seqs = len(self.src_seqs)
        self.src_word2id = src_word2id
        self.trg_word2id = trg_word2id

    def __getitem__(self, index):
        """Returns one data pair (source and target)."""
        src_seq = self.src_seqs[index]
        trg_seq = self.trg_seqs[index]
        src_seq = self.preprocess(src_seq, self.src_word2id, trg=False)
        trg_seq = self.preprocess(trg_seq, self.trg_word2id)
        return src_seq, trg_seq

    def __len__(self):
        return self.num_total_seqs

def preprocess(self, sequence, word2id, trg=True):
    """Converts words to ids."""
    tokens = nltk.tokenize.word_tokenize(sequence.lower())
    sequence = []
    sequence.append(word2id['<start>'])
    sequence.extend([word2id[token] for token in tokens if token in word2id])
    sequence.append(word2id['<end>'])
    sequence = torch.Tensor(sequence)
    return sequence
```

https://github.com/yunjey/seq2seq-dataloader/blob/master/data_loader.py

Data Preprocessing

- Define operations on Dataset
  - call the operations in `__getitem__`
  - or use built-in functions

```python
class CustomDataset(Dataset, transform=None):
    """
    Skeleton of Custom Dataset
    Custom Dataset 의 기초 골격
    """

    def __init__(self):
        """
        Initial setting: reading a csv, assigning transforms etc.
        데이터셋 초기화. csv 파일 읽는 부분 또는 전처리를 담당
        """
        self.transform = transform

    def __len__(self):
        """
        Returns the size of the dataset
        데이터셋의 길이
        """

    def __getitem__ (self, idx):
        """
        Extract one sample from a dataset.
        데이터셋에서 1개의 sample 을 추출.
        """

        if self.transform:
            x = self.transform(x)
```

```python
from torchvision import transforms

data_transform = transforms.Compose([
    transforms.RandomSizedCrop(224),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406],
                         std=[0.229, 0.224, 0.225])
])

# Apply the above transform to CustomDataset
dataset = CustomDataset(..., transform=data_transform)
```

Data Preprocessing

- Define operations on Dataset
  - call the operations in `__getitem__`
  - or use built-in functions

```python
from torchvision import transforms

data_transform = transforms.Compose([
    transforms.RandomSizedCrop(224),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406],
                         std=[0.229, 0.224, 0.225])
])

# Apply the above transform to CustomDataset
dataset = CustomDataset(..., transform=data_transform)
```

```python
class CustomDataset(Dataset, transform=None):
    """
    Skeleton of Custom Dataset
    Custom Dataset 의 기초 골격
    """

    def __init__(self):
        """
        Initial setting: reading a csv, assigning transforms etc.
        데이터셋 초기화. csv 파일 읽는 부분 또는 전처리를 담당
        """
        self.transform = transform

    def __len__(self):
        """
        Returns the size of the dataset
        데이터셋의 길이
        """

    def __getitem__ (self, idx):
        """
        Extract one sample from a dataset.
        데이터셋에서 1개의 sample 을 추출.
        """

        if self.transform:
            x = self.transform(x)
```

K-fold Cross-Validation?

- Various implementation is possible
  - construct two custom datasets `train_dataset` `valid_dataset`
  - supply pre-defined index sets to datasets
  - reference: link
- Use SKORCH
  - make it possible to use PyTorch with sklearn
  - one can implement `StratifiedKFold`

I want to utilize *new* functions..

I have a great idea!

- Suppose you have an idea about
  - new activation functions
  - new loss functions
  - new modules
- How can I *implement* custom functions?
  - if you can implement those functions in `NumPy`, then it is also possible for `PyTorch`

Auto Differentiation

- Compute the gradient of f with respect to x, y, z

$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$

I love calculus!

Auto Differentiation

- Compute the gradient of f with respect to x, y, z

$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$

```
In[1]:= f[x_, y_, z_] := x Sqrt[Exp[-Sin[z]/y^2]]
        제곱근  지수 함수
```

symbolic calculus
is good but slow

```
In[5]:= {D[f[x, y, z], x], D[f[x, y, z], y], D[f[x, y, z], z]}
        미분 계수            미분 계수            미분 계수
```

$$\text{Out[5]} = \left\{\sqrt{e^{-\frac{\sin[z]}{y^2}}},\ \frac{\sqrt{e^{-\frac{\sin[z]}{y^2}}}\ x \sin[z]}{y^3},\ -\frac{\sqrt{e^{-\frac{\sin[z]}{y^2}}}\ x \cos[z]}{2y^2}\right\}$$

Auto Differentiation

- Compute the gradient of f with respect to x, y, z

$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$

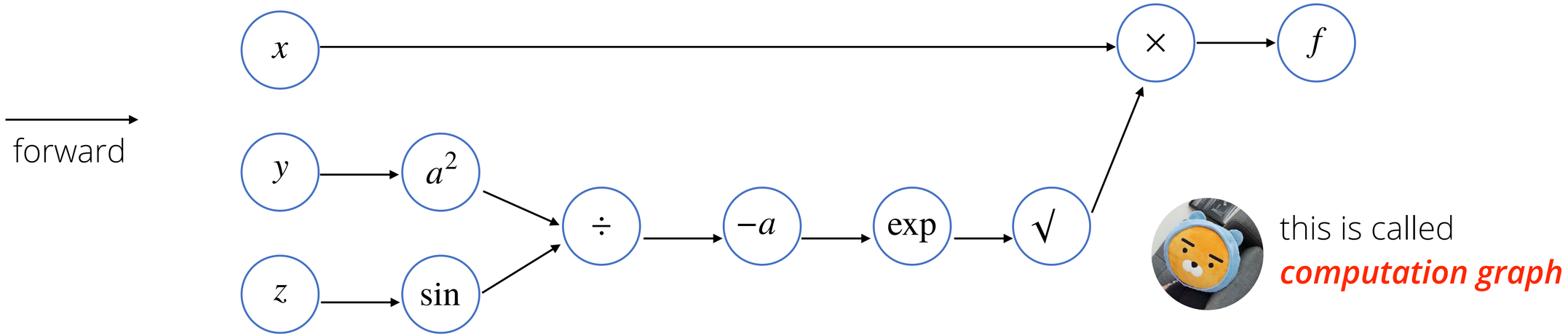
$$\left(\frac{f(x+\epsilon, y, z) - f(x, y, z)}{\epsilon}, \frac{f(x, y+\epsilon, z) - f(x, y, z)}{\epsilon}, \frac{f(x, y, z+\epsilon) - f(x, y, z)}{\epsilon}\right)$$

but difference method
accumulates errors gradually

Auto Differentiation

- Compute the gradient of f with respect to x, y, z

$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$

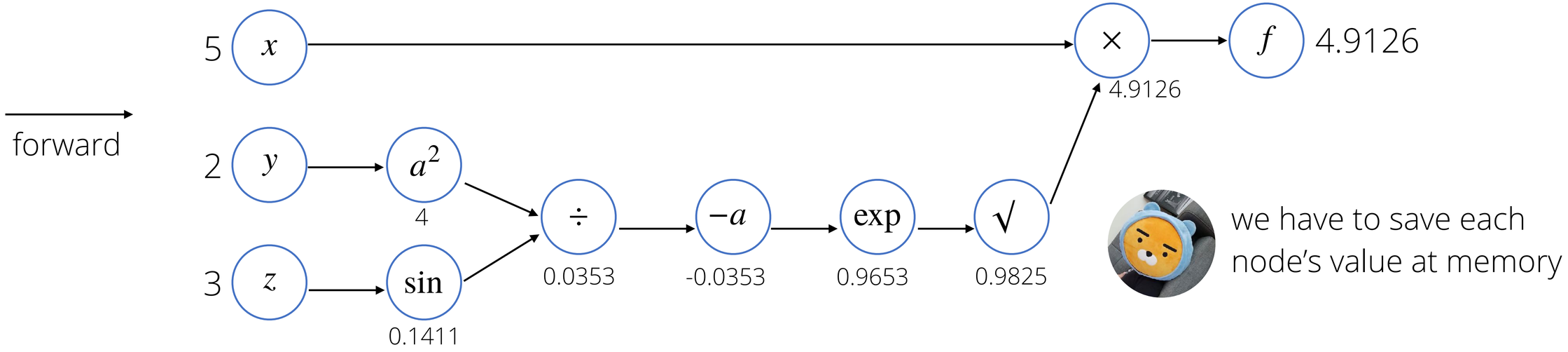


this is called
computation graph

Auto Differentiation

- Compute the gradient of f with respect to x, y, z

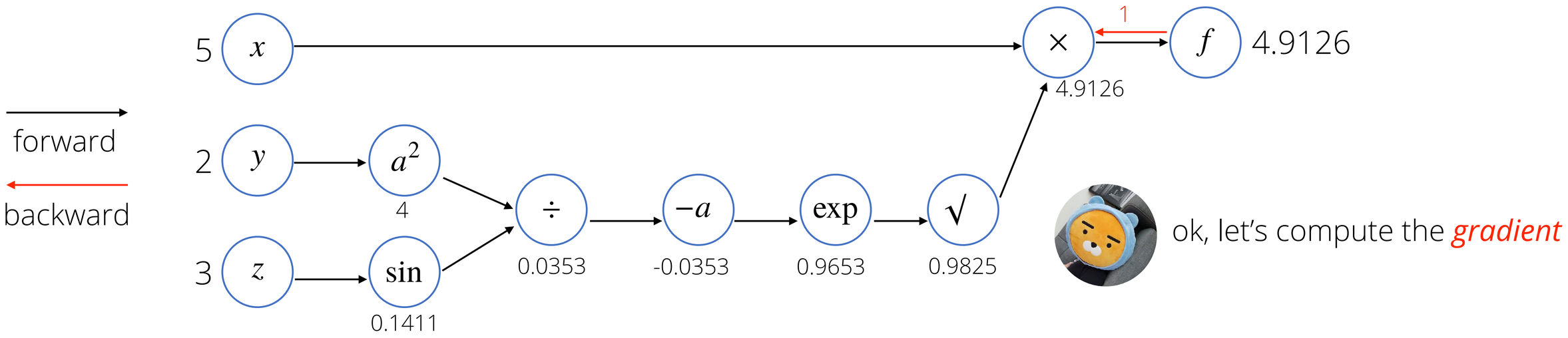
$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



Auto Differentiation

- Compute the gradient of f with respect to x, y, z

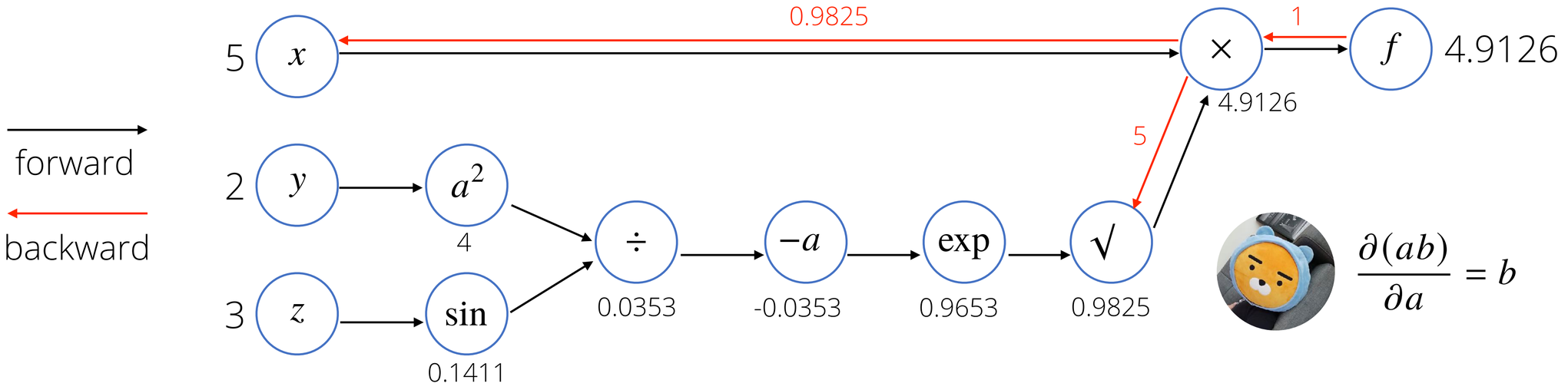
$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



Auto Differentiation

- Compute the gradient of f with respect to x, y, z

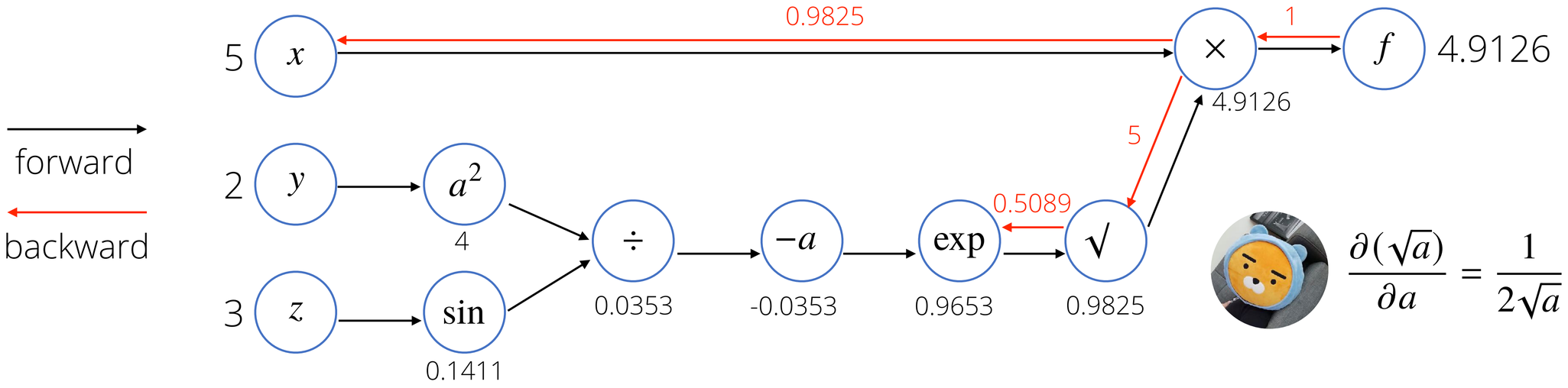
$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



Auto Differentiation

- Compute the gradient of f with respect to x, y, z

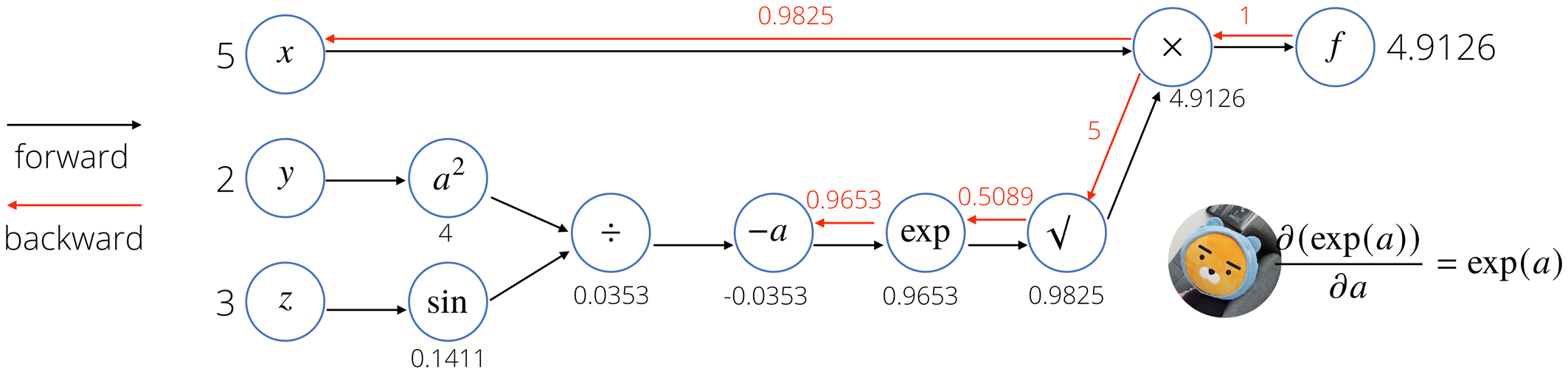
$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



Auto Differentiation

- Compute the gradient of f with respect to x, y, z

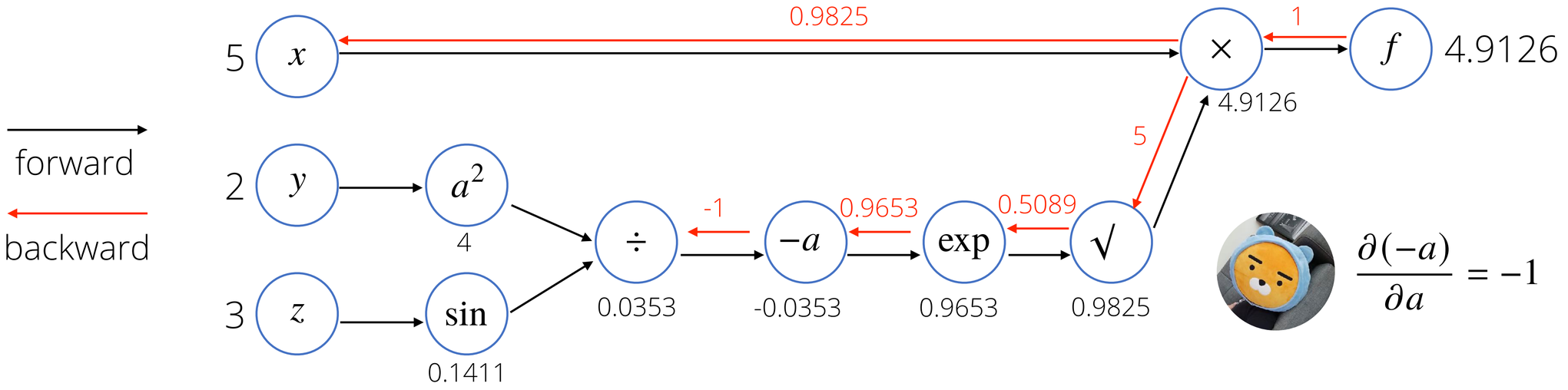
$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



Auto Differentiation

- Compute the gradient of f with respect to x, y, z

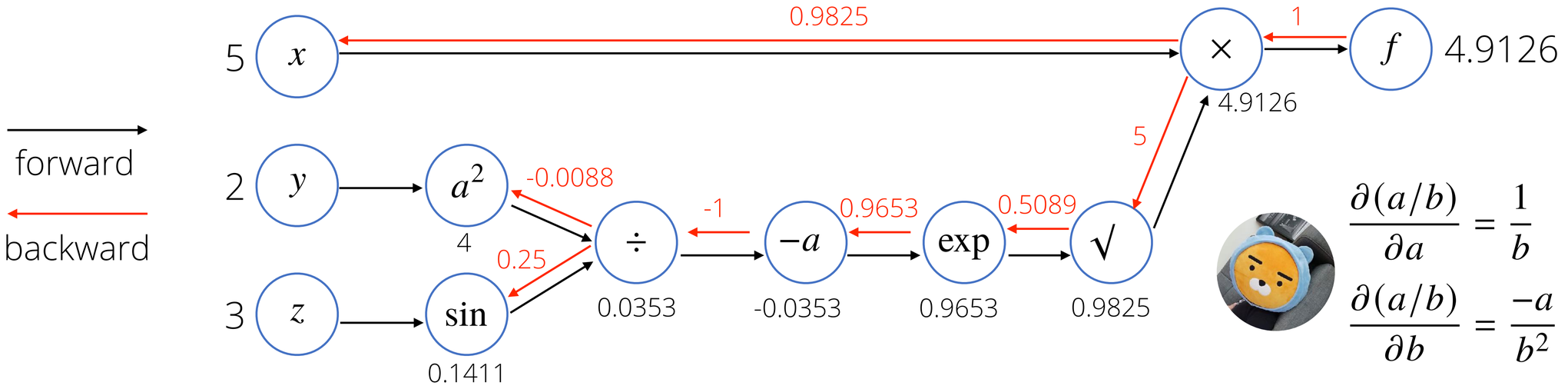
$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



Auto Differentiation

- Compute the gradient of f with respect to x, y, z

$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



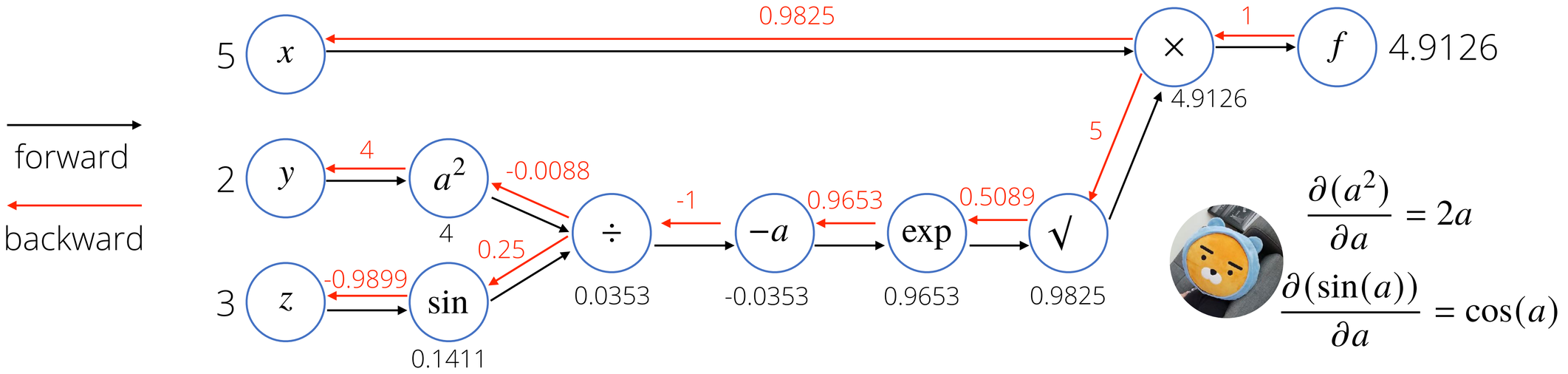
$$\frac{\partial(a/b)}{\partial a} = \frac{1}{b}$$

$$\frac{\partial(a/b)}{\partial b} = \frac{-a}{b^2}$$

Auto Differentiation

- Compute the gradient of f with respect to x, y, z

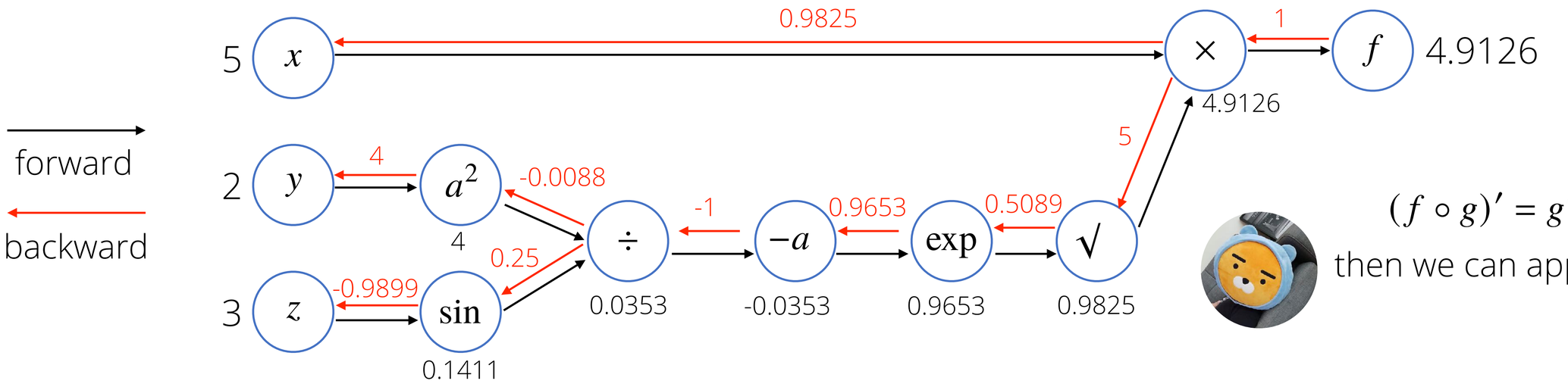
$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



Auto Differentiation

- Compute the gradient of f with respect to x, y, z

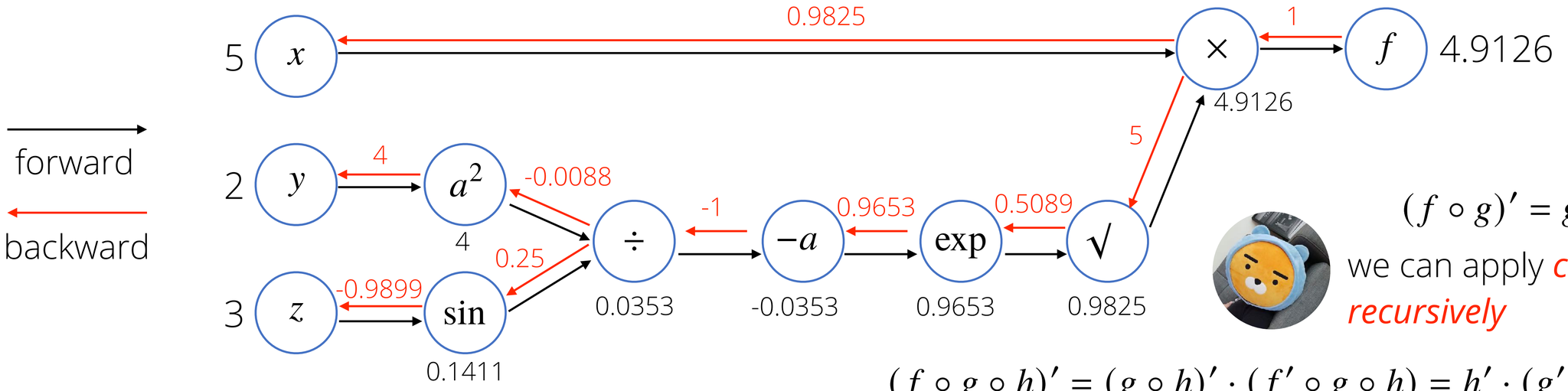
$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



Auto Differentiation

- Compute the gradient of f with respect to x, y, z

$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



$(f \circ g)' = g' \cdot (f' \circ g)$

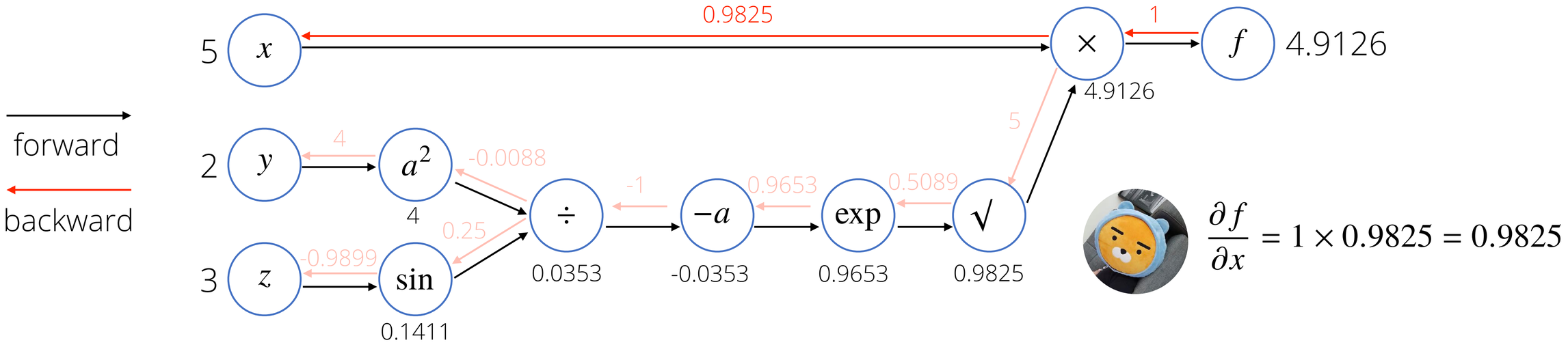
we can apply *chain rule recursively*

$$(f \circ g \circ h)' = (g \circ h)' \cdot (f' \circ g \circ h) = h' \cdot (g' \circ h) \cdot (f' \circ g \circ h)$$

Auto Differentiation

- Compute the gradient of f with respect to x, y, z

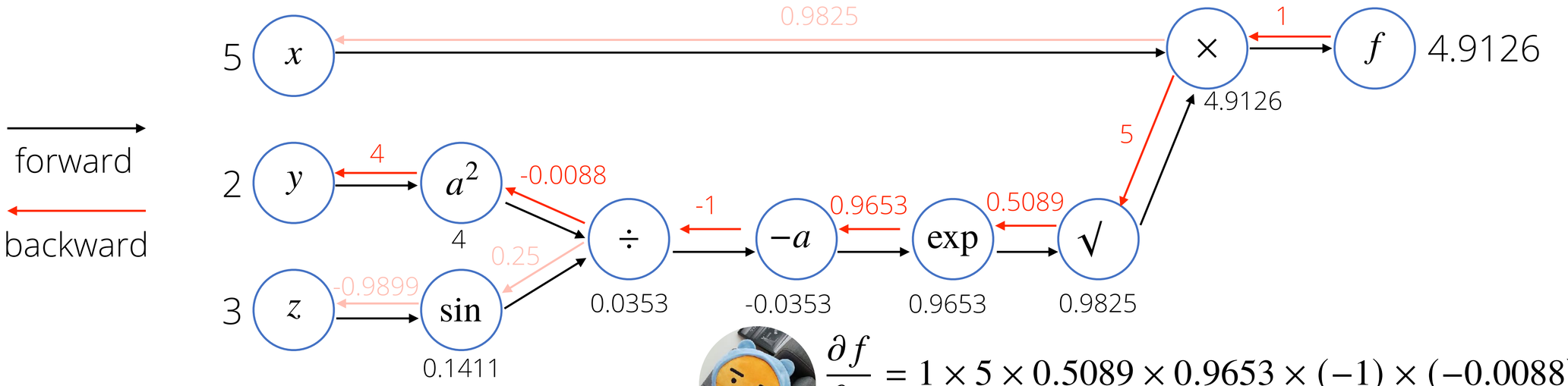
$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



Auto Differentiation

- Compute the gradient of f with respect to x, y, z

$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$

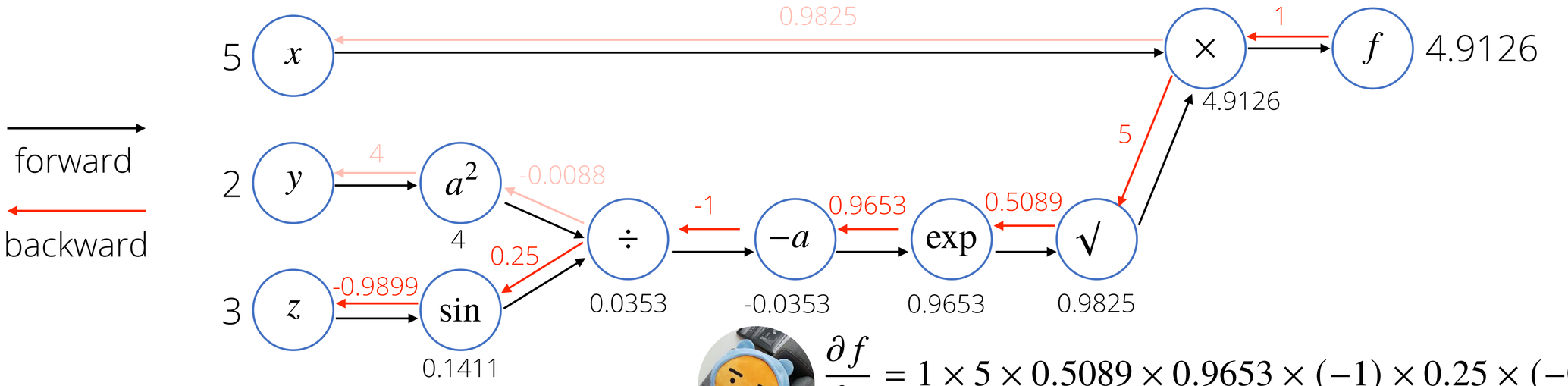


$$\frac{\partial f}{\partial y} = 1 \times 5 \times 0.5089 \times 0.9653 \times (-1) \times (-0.0088) \times 4 = 0.0865$$

Auto Differentiation

- Compute the gradient of f with respect to x, y, z

$$f(x, y, z) = x\sqrt{\exp\left(-\frac{\sin z}{y^2}\right)} \qquad \nabla f = (\partial_x f, \partial_y f, \partial_z f) = ?$$



$$\frac{\partial f}{\partial z} = 1 \times 5 \times 0.5089 \times 0.9653 \times (-1) \times 0.25 \times (-0.9899) = 0.6079$$

Computation Graph

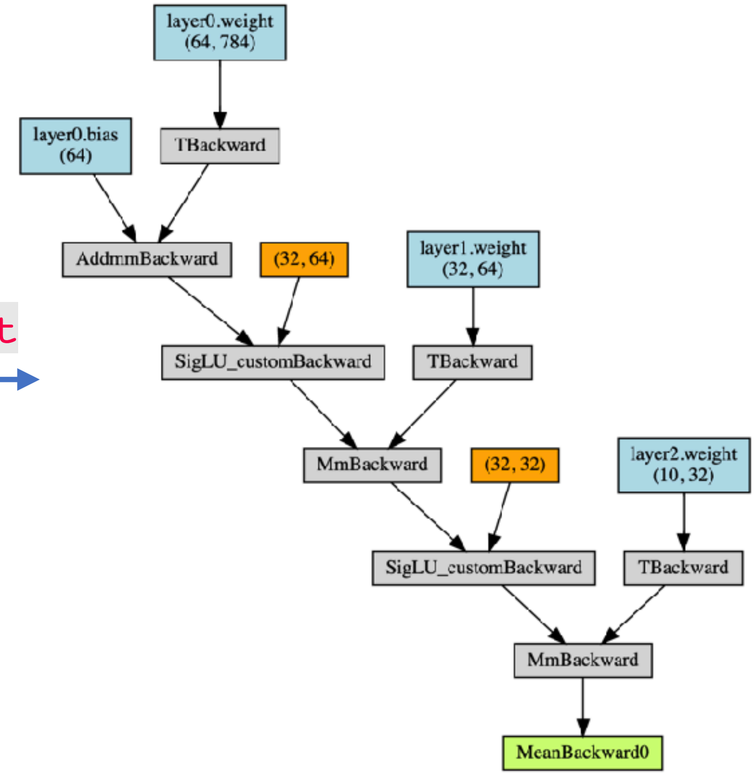
- How can I plot the computation graph? → use `torchviz`

```python
class MLP_SigLU(nn.Module):
    def __init__(self):
        super(MLP_SigLU, self).__init__()
        self.layer0 = torch.nn.Linear(28*28, 64, bias=True)
        # self.act = torch.nn.ReLU()
        self.act = SigLU_custom()
        self.layer1 = torch.nn.Linear(64, 32, bias=False)
        self.layer2 = torch.nn.Linear(32, 10, bias=False)

    def forward(self, inputs):
        inputs = inputs.view(-1, 28*28) # match dimension
        hidden = self.layer0(inputs)
        hidden = self.act.apply(hidden)
        hidden = self.layer1(hidden)
        hidden = self.act.apply(hidden)
        outputs = self.layer2(hidden)
        return outputs

model = MLP_SigLU()
```

`torchviz.make_dot` →



Checkpoints

- How to save parameters in the training? → use `torch.save`

```
torch.save(model.state_dict(), PATH)
```

```
model = TheModelClass(*args, **kwargs)
model.load_state_dict(torch.load(PATH))
```

very simple!

Assignments

- Revise your proposal in detail (until 4/9) → Determine topics (until 4/23)
  - Explain your Data Science problems in detail
    - What is your goal? Why this is important problem?
    - How to evaluate your model empirically? What is your metric?
  - Describe [what is, how to gather, difficulty of] your data concretely
  - Read the issues in your GitHub and revise your proposal

begin your project

begin your project and complement proposal

complement proposal ***before*** begin your project,

Assignments

- Read papers
  - LeNet (1998): Gradient-Based Learning Applied to Document Recognition
  - AlexNet (2012): ImageNet Classification with Deep Convolutional Neural Networks

  until 4/9

  - ZFNet (2013): Visualizing and Understanding Convolutional Networks
  - VGG (2014): Very Deep Convolutional Networks for Large-Scale Image Recognition
  - Inception (2014): Going Deeper with Convolutions
  - ResNet (2015): Deep Residual Learning for Image Recognition
  - WideResNet (2016): Wide Residual Networks
  - DenseNet (2016): Densely Connected Convolutional Networks

  until 4/14

  - NAS (2018): Learning Transferable Architectures for Scalable Image Recognition
  - EfficientNet (2019): Rethinking Model Scaling for Convolutional Neural Networks

  until 4/16

Q & A