BITS Pilani

Pilani | Dubai | Goa | Hyderabad

S1-25_DSECLZG530/SSCLZG599
Natural Language Processing
(Lecture #5 – LLMs, Prompt Engg)

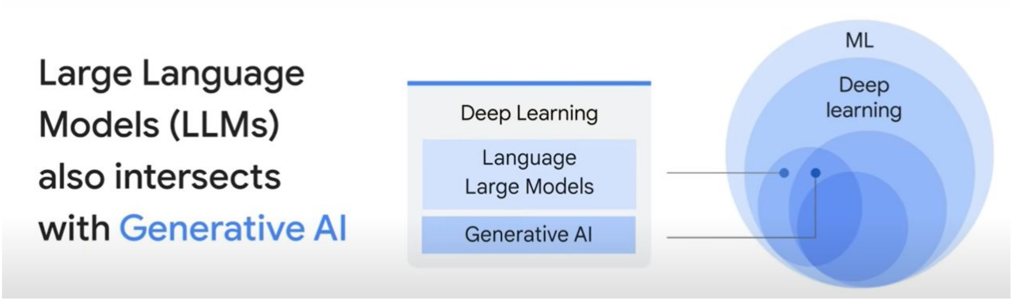
- *The slides presented here are obtained from the authors of the books and from various other contributors. I hereby acknowledge all the contributors for their material and inputs.*
- *I have added and modified a few slides to suit the requirements of the course.*

Session Content

Neural Networks and Neural Language Models

- LLMs
- Prompt Engineering
- Computation Graphs & Backward Differentiation

Large Language Models



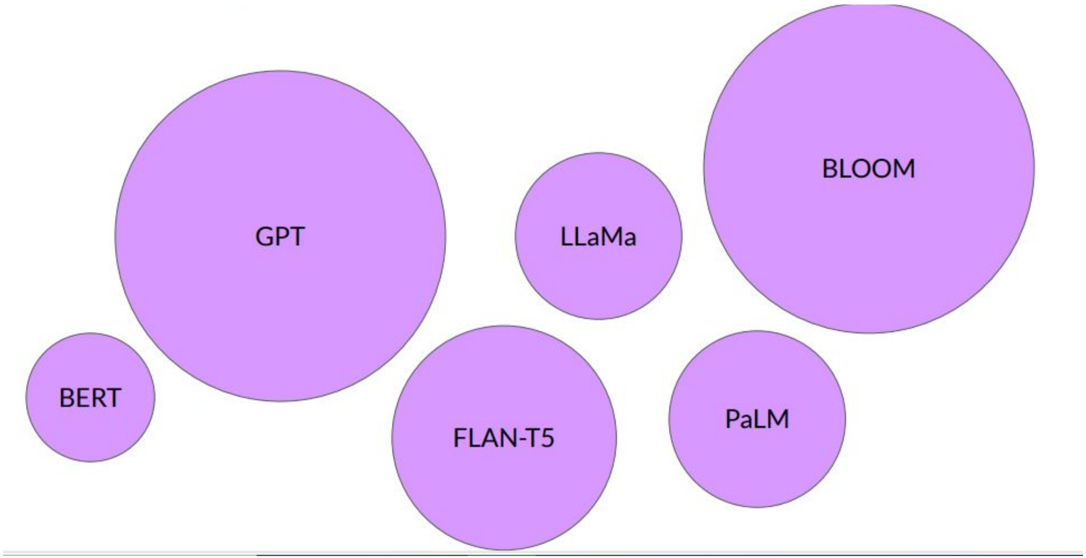
Large Language Models

Large, general-purpose language models can be pre-trained and then fine-tuned for specific purposes

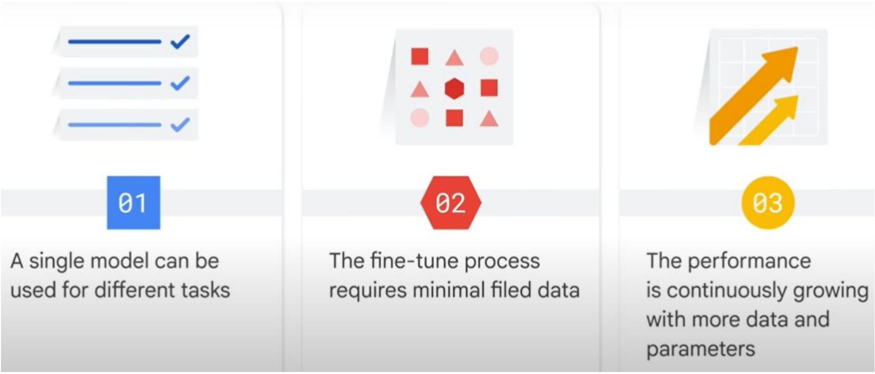
Large Language Models

- **Large**
  - Large training dataset
  - Large number of parameters
- **General purpose**
  - Commonality of human languages
  - Resource restriction
- **Pre-trained and fine-tuned**

Large Language Models

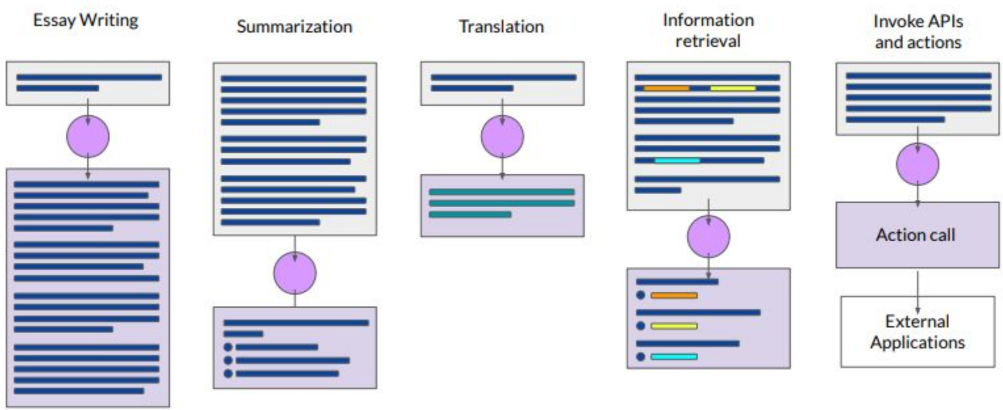


Benefits of using Large Language Models

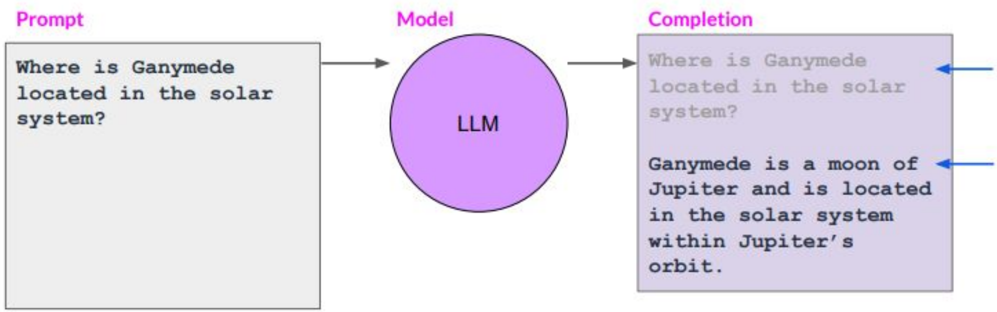


LLM Development (using pre-trained APIs)	Traditional ML Development
• NO ML expertise needed • NO training examples • NO need to train a model • Thinks about prompt design	• YES ML expertise needed • YES training examples • YES need to train a model • YES compute time + + hardware • Thinks about minimizing a loss function

LLM Use Cases



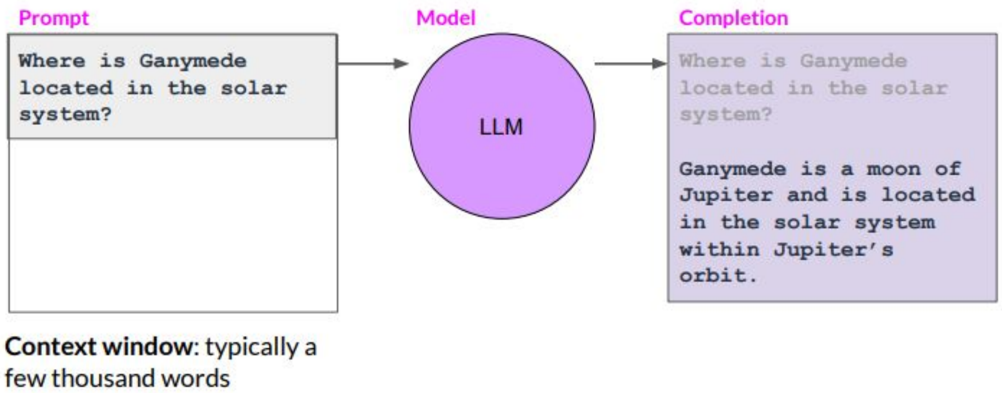
Prompts and Completions



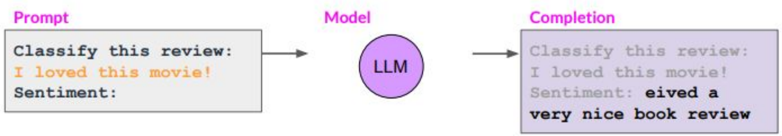
Context window

- typically a few 1000 words.

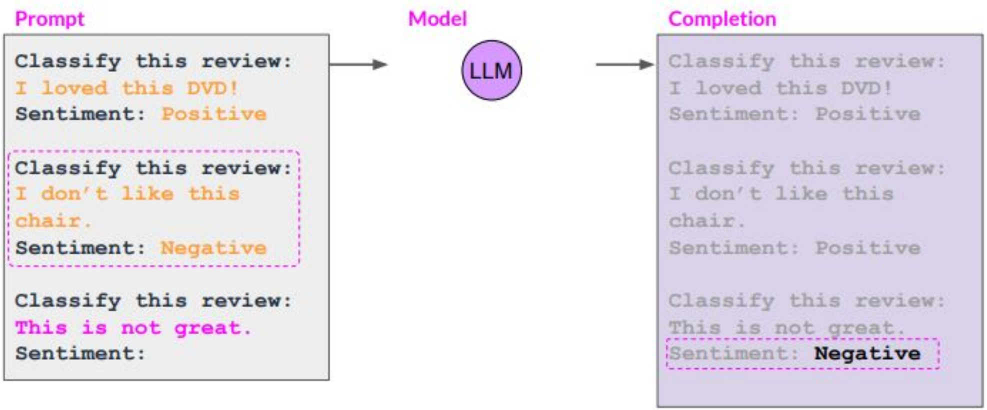
Prompting and Prompt Engineering



In context learning and zero shot inference



In context learning and Few shot inference



DL Model Types

Deep Learning Model Types

Discriminative

- Used to classify or predict
- Typically trained on a dataset of labeled data
- Learns the relationship between the features of the data points and the labels

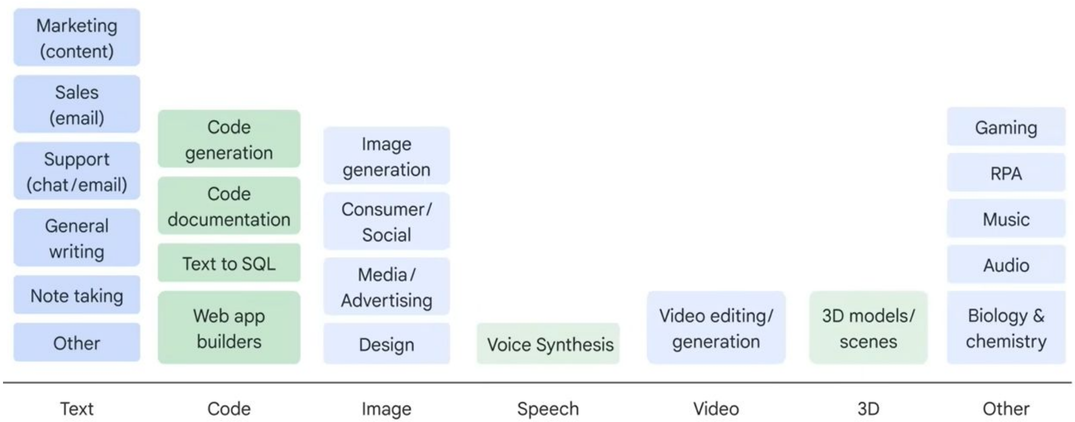
Generative

- Generates new data that is similar to data it was trained on
- Understands distribution of data and how likely a given example is
- Predict next word in a sequence

Generative AI

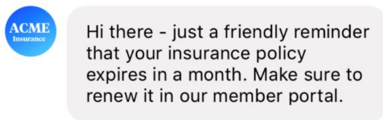
- GenAI is a type of Artificial Intelligence that creates new content based on what it has learned from existing content.
- The process of learning from existing content is called training and results in the creation of a statistical model.
- When given a prompt, GenAI uses this statistical model to predict what an expected response might be–and this generates new content.

Generative AI Applications

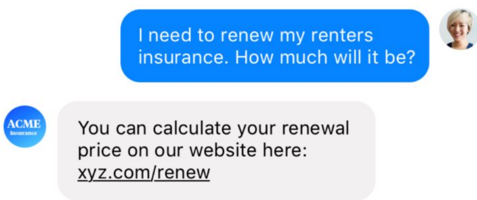


Generative AI- AI Assistants

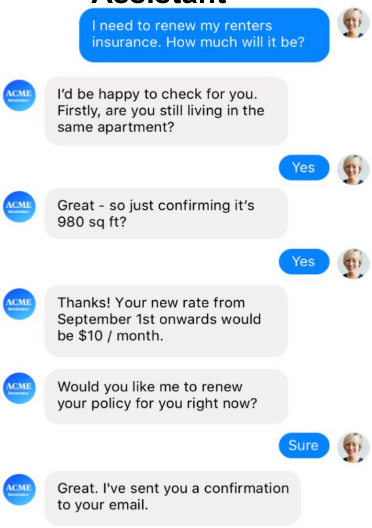
Notification Assistant



FAQ Assistant



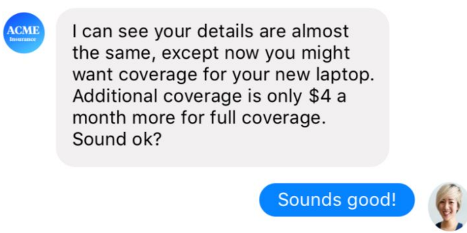
Contextual Assistant



Generative AI- AI Assistants

Personalized Assistant

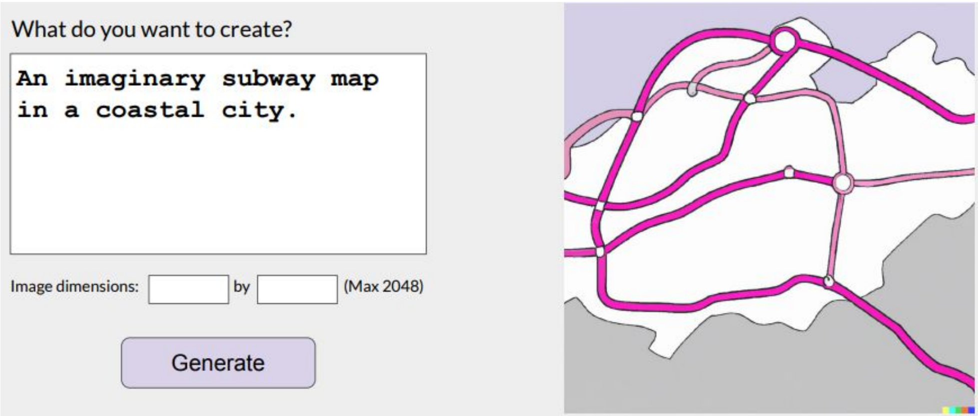
- Assistant knows you much more in detail
- Quickly checks a few final things before giving you a quote tailored to your actual situation.



Autonomous Organization of Assistants

- Group of AI assistants that know every customer personally
- Eventually run large parts of company operations—from lead generation over marketing, sales, HR, or finance

Generative AI- PaintBox



Can you create an imaginary subway map for a coastal city?

Gemini:

Generative AI- CodeAid

```
def binary_search(arr, x, l, r):_
    if r >= l:
        mid = l + (r - l) // 2
        if arr[mid] == x:
            return mid
        elif arr[mid] > x:
            return binary_search(arr, x, l, mid - 1)
        else:
            return binary_search(arr, x, mid + 1, r)
    else:
        return -1
```

< 1/2 > Accept Tab

Real-life challenges in NLP tasks

- Deep learning methods are data-hungry
- >50K data items needed for training
- The distributions of the source and target data must be the same
- Labeled data in the target domain may be limited
- This problem is typically addressed with **transfer learning**

Transfer Learning

- Using a pre-trained model as a starting point for a new task or domain.
- Leverage the knowledge acquired by the pre-trained model on a large dataset and apply it to a related task with a smaller dataset.
- We can benefit from the general features and patterns learned by the pre-trained model, saving time and computational resources.
- Transfer learning involves freezing the pre-trained model's weights and only training the new layers

Ex: image classification, knowledge gained while learning to recognize cars could be applied when trying to recognize trucks

Transductive vs Inductive Transfer Learning

- **Transductive** transfer
  - No labeled target domain data available
  - Focus of most transfer research in NLP
  - E.g. Domain adaptation
- **Inductive** transfer
  - Labeled target domain data available
  - Goal: improve performance on the target task by training on other task(s)
    - Jointly training on >1 task (multi-task learning)
    - Pre-training (e.g. word embeddings)

Applications of Transfer Learning

- Image Classification
- Names Entity Recognition
- Sentiment Analysis
- Cross Lingual Learning
- Gaming
- Image Recognition
- Speech Recognition

Fine Tuning

- Fine-tuning takes it a step further by allowing the pre-trained layers to be updated.

- Beneficial when the new dataset is large enough and similar to the original dataset on which the pre-trained model was trained

References

CH-7 - Speech and Language Processing by Daniel Jurafsky https://www.youtube.com/watch?v=BtmsIy0j_dY&t=5s

innovate achieve lead

BITS Pilani

Pilani | Dubai | Goa | Hyderabad

Computation Graphs

Why Computation Graphs

- For training, we need the derivative of the loss with respect to each weight in every layer of the network
- But the loss is computed only at the very end of the network!
- Solution: **error backpropagation** (Rumelhart, Hinton, Williams, 1986)
- **Backprop** is a special case of **backward differentiation**
- Which relies on **computation graphs**.

Computation Graphs

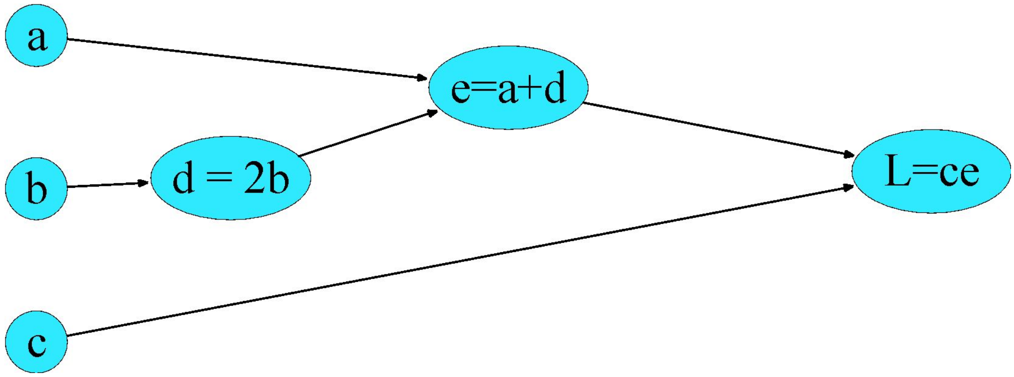
- A computation graph represents the process of computing a mathematical expression

Example:

$$L(a,b,c) = c(a+2b)$$

Computations:

$$\begin{aligned} d &= 2*b \\ e &= a+d \\ L &= c*e \end{aligned}$$

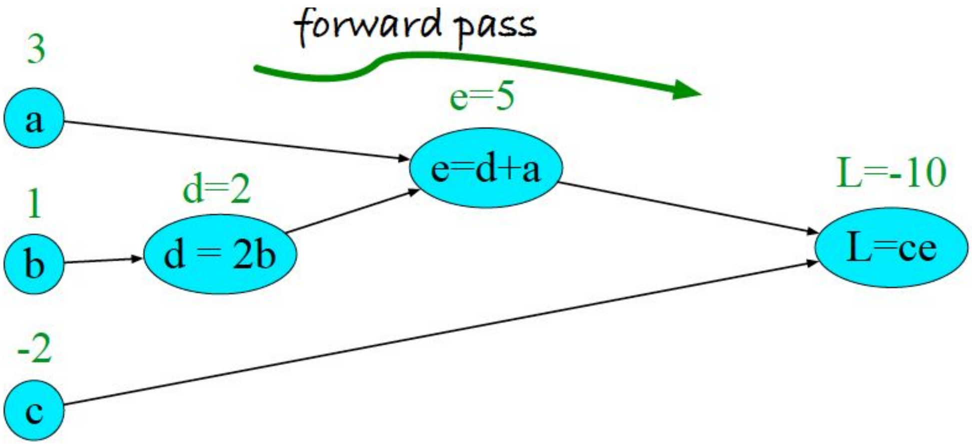


Example:

$$L(a,b,c) = c(a+2b)$$

Computations:

$$\begin{aligned} d &= 2*b \\ e &= a+d \\ L &= c*e \end{aligned}$$



Backwards differentiation in computation graphs

- The importance of the computation graph comes from the backward pass
- This is used to compute the derivatives that we'll need for the weight update.

Example:

$$L(a,b,c) = c(a+2b)$$

$$\begin{aligned} d &= 2*b \\ e &= a+d \\ L &= c*e \end{aligned}$$

We want: $\frac{\partial L}{\partial a}$, $\frac{\partial L}{\partial b}$, and $\frac{\partial L}{\partial c}$

The derivative $\frac{\partial L}{\partial a}$, tells us how much a small change in *a* affects *L*.

The chain rule

- Computing the derivative of a composite function:
- $f(x) = u(v(x))$ $$\frac{df}{dx} = \frac{du}{dv} \cdot \frac{dv}{dx}$$
- $f(x) = u(v(w(x)))$ $$\frac{df}{dx} = \frac{du}{dv} \cdot \frac{dv}{dw} \cdot \frac{dw}{dx}$$

Example

$$L(a,b,c) = c(a+2b)$$

$$\begin{aligned} d &= 2*b \\ e &= a+d \\ L &= c*e \end{aligned}$$

$$\begin{aligned} \frac{\partial L}{\partial c} &= \epsilon \\ \frac{\partial L}{\partial a} &= \frac{\partial L}{\partial e}\frac{\partial e}{\partial a} \\ \frac{\partial L}{\partial b} &= \frac{\partial L}{\partial e}\frac{\partial e}{\partial d}\frac{\partial d}{\partial b} \end{aligned}$$

Example

$$L(a,b,c) = c(a+2b)$$

$$\begin{aligned} d &= 2*b \\ e &= a+d \\ L &= c*e \end{aligned}$$

$$\begin{aligned} \frac{\partial L}{\partial a} &= \frac{\partial L}{\partial e}\frac{\partial e}{\partial a} \\ \frac{\partial L}{\partial b} &= \frac{\partial L}{\partial e}\frac{\partial e}{\partial d}\frac{\partial d}{\partial b} \end{aligned}$$

$$\begin{aligned} L = ce &: \quad \frac{\partial L}{\partial e} = c, \frac{\partial L}{\partial c} = e \\ e = a+d &: \quad \frac{\partial e}{\partial a} = 1, \frac{\partial e}{\partial d} = 1 \\ d = 2b &: \quad \frac{\partial d}{\partial b} = 2 \end{aligned}$$

Example

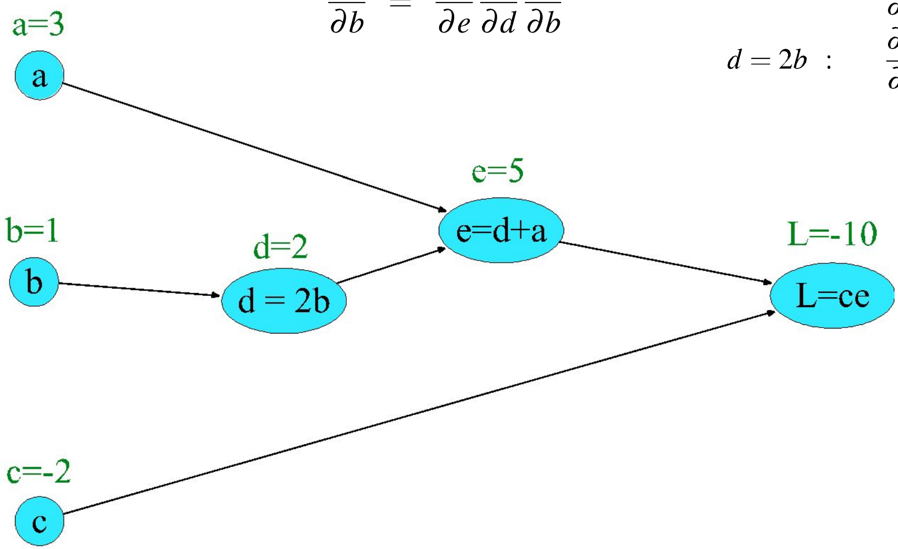
$$\frac{\partial L}{\partial a} = \frac{\partial L}{\partial e}\frac{\partial e}{\partial a}$$

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial e}\frac{\partial e}{\partial d}\frac{\partial d}{\partial b}$$

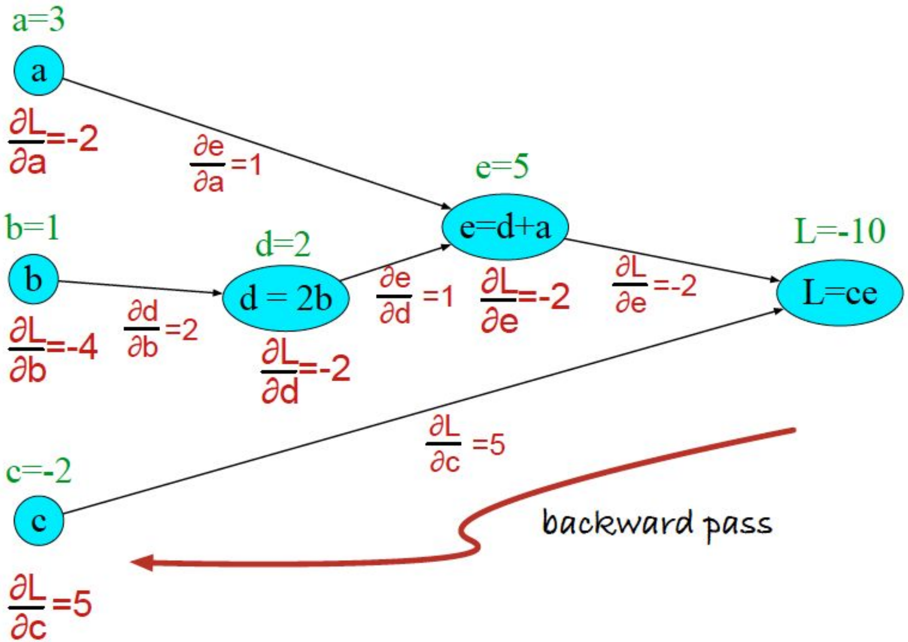
$$L = ce \ : \quad \frac{\partial L}{\partial e} = c, \frac{\partial L}{\partial c} = e$$

$$e = a + d \ : \quad \frac{\partial e}{\partial a} = 1, \frac{\partial e}{\partial d} = 1$$

$$d = 2b \ : \quad \frac{\partial d}{\partial b} = 2$$

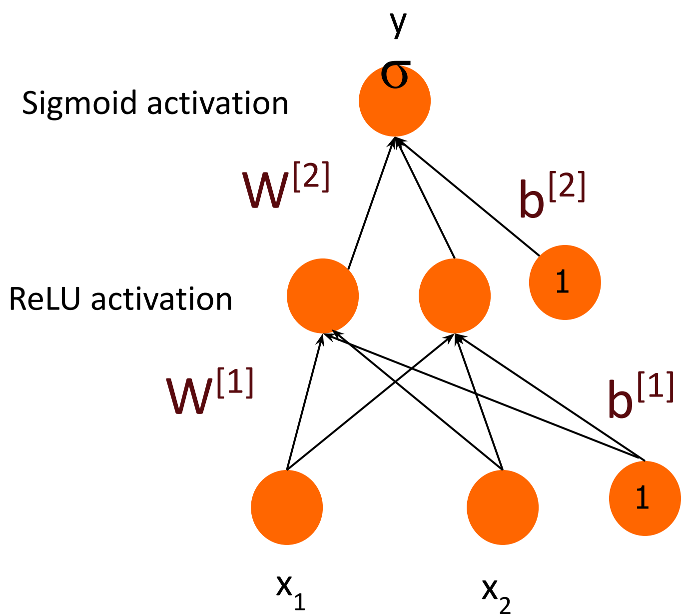


Example



a=3
a
$\frac{\partial L}{\partial a}$=-2
$\frac{\partial e}{\partial a}$=1
e=5
e=d+a
b=1
b
d=2
d = 2b
$\frac{\partial e}{\partial d}$=1
$\frac{\partial L}{\partial e}$=-2
$\frac{\partial L}{\partial e}$=-2
L=-10
L=ce
$\frac{\partial d}{\partial b}$=2
$\frac{\partial L}{\partial b}$=-4
$\frac{\partial L}{\partial d}$=-2
$\frac{\partial L}{\partial c}$=5
c=-2
c
backward pass
$\frac{\partial L}{\partial c}$=5

Backward differentiation on a two layer network



$$
\begin{aligned}
z^{[1]} &= W^{[1]}\mathbf{x} + b^{[1]} \\
a^{[1]} &= \mathrm{ReLU}(z^{[1]}) \\
z^{[2]} &= W^{[2]}a^{[1]} + b^{[2]} \\
a^{[2]} &= \sigma(z^{[2]}) \\
\hat{y} &= a^{[2]}
\end{aligned}
$$

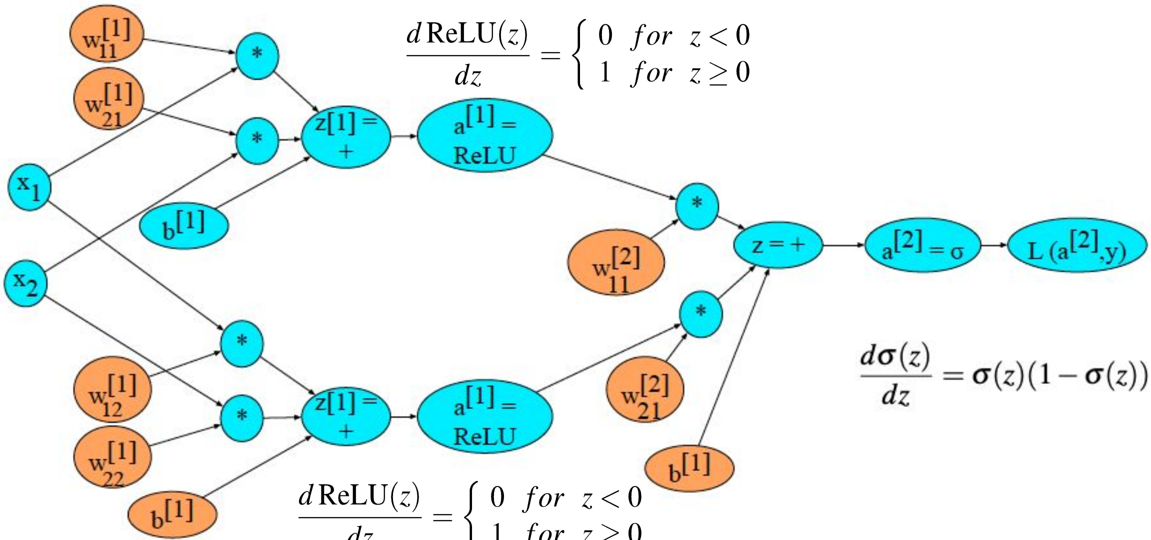
Backward differentiation on a two layer network

$$
\begin{aligned}
z^{[1]} &= W^{[1]}\mathbf{x} + b^{[1]} \\
a^{[1]} &= \mathrm{ReLU}(z^{[1]}) \\
z^{[2]} &= W^{[2]}a^{[1]} + b^{[2]} \\
a^{[2]} &= \sigma(z^{[2]}) \\
\hat{y} &= a^{[2]}
\end{aligned}
$$

$$
\frac{d\,\mathrm{ReLU}(z)}{dz} = \begin{cases} 0 & for \quad z < 0 \\ 1 & for \quad z \geq 0 \end{cases}
$$

$$
\frac{d\sigma(z)}{dz} = \sigma(z)(1 - \sigma(z))
$$

Backward differentiation on a 2-layer network



$$\frac{d\,\mathrm{ReLU}(z)}{dz} = \begin{cases} 0 & for \quad z < 0 \\ 1 & for \quad z \geq 0 \end{cases}$$

$$\frac{d\sigma(z)}{dz} = \sigma(z)(1-\sigma(z))$$

$$\frac{d\,\mathrm{ReLU}(z)}{dz} = \begin{cases} 0 & for \quad z < 0 \\ 1 & for \quad z \geq 0 \end{cases}$$

Starting off the backward pass: $\frac{\partial L}{\partial z}$

(I'll write a for $a^{[2]}$ and z for $z^{[2]}$)

$$
\begin{aligned}
z^{[1]} &= W^{[1]}\mathbf{x} + b^{[1]} \\
a^{[1]} &= \mathrm{ReLU}(z^{[1]}) \\
z^{[2]} &= W^{[2]}a^{[1]} + b^{[2]} \\
a^{[2]} &= \sigma(z^{[2]}) \\
\hat{y} &= a^{[2]}
\end{aligned}
$$

$$L(\hat{y}, y) = -(\mathrm{y} \log(\hat{y}) + (1-y)\log(1-\hat{y}))$$

$$L(a, y) = -(\mathrm{y} \log a + (1-y)\log(1-a))$$

$$\frac{\partial L}{\partial z} = \frac{\partial L}{\partial a}\frac{\partial a}{\partial z}$$

$$\frac{\partial L}{\partial a} = -\left(\left(y\frac{\partial \log(a)}{\partial a}\right) + (1-\mathrm{y})\,\frac{\partial \log(1-a)}{\partial a}\right)$$

$$= -\left(\left(y\frac{1}{a}\right) + (1-\mathrm{y})\,\frac{1}{1-a}(-1)\right) = -\left(\frac{y}{a} + \frac{y-1}{1-a}\right)$$

$$\frac{\partial a}{\partial z} = a(1-a) \qquad \frac{\partial L}{\partial z} = -\left(\frac{y}{a} + \frac{y-1}{1-a}\right)a(1-a) = \mathrm{a} - \mathrm{y}$$

Thank You

References

	Author(s), Title, Edition, Publishing House
T1	Speech and Language processing: An introduction to Natural Language Processing, Computational Linguistics and speech Recognition by Daniel Jurafsky and James H. Martin[3rd edition]
T2	Foundations of statistical Natural language processing by Christopher D.Manning and Hinrich schutze
	Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing https://arxiv.org/pdf/2107.13586

BITS Pilani, Pilani Campus