deeplearning.ai

Overview

Learning objectives

Learning objectives

- What is autocorrect?

Learning objectives

- What is autocorrect?

- Building the model

deah → dear ✅

yeah

dear

dean

... *etc*

Learning objectives

- What is autocorrect?
- Building the model
- Minimum edit distance

deah → dear ✅

yeah

dear

dean

... *etc*

	#	s	t	a	y
#	0	1	2	3	4
p	1	2	3	4	5
l	2	3	4	5	6
a	3	4	5	4	5
y	4	5	6	5	4

Learning objectives

- What is autocorrect?
- Building the model
- Minimum edit distance
- Minimum edit distance algorithm

deah → dear ✅
yeah
dear
dean
... *etc*

	#	s	t	a	y
#	0	1	2	3	4
p	1	2	3	4	5
l	2	3	4	5	6
a	3	4	5	4	5
y	4	5	6	5	4

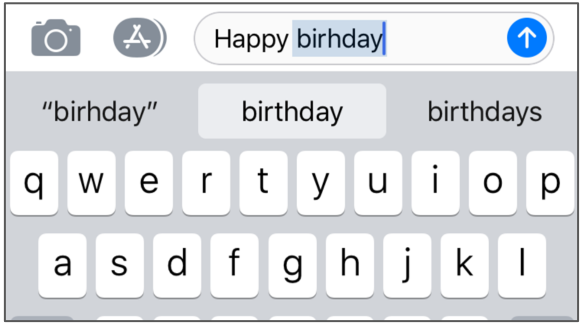
deeplearning.ai

Autocorrect

What is autocorrect?

What is autocorrect?

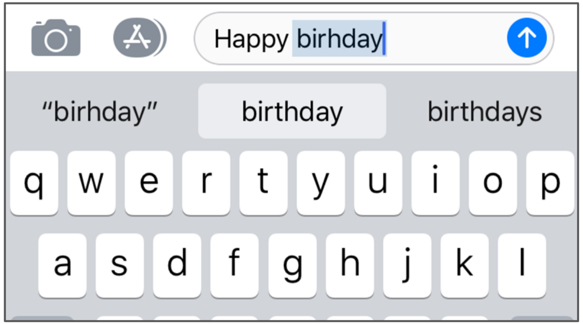
- Phones



Happy birhday

What is autocorrect?

- Phones
- Tablets

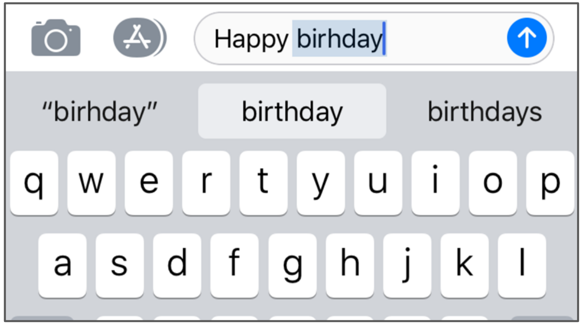


Happy birhday

“birhday” | birthday | birthdays

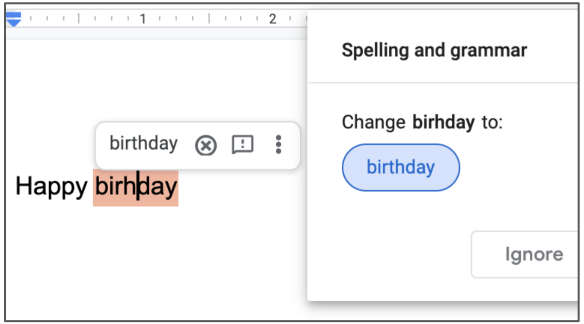
What is autocorrect?

- Phones
- Tablets
- Computers



Happy birhday

"birhday" | birthday | birthdays



birthday

Happy birhday

Spelling and grammar

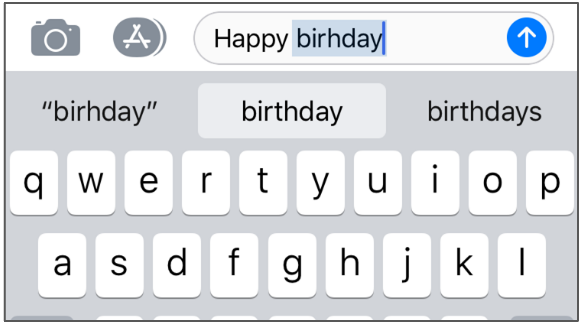
Change **birhday** to:

birthday

Ignore

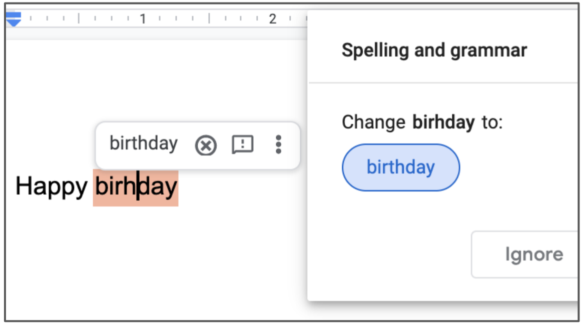
What is autocorrect?

- Phones
- Tablets
- Computers



Happy birhday

"birhday" birthday birthdays



birthday

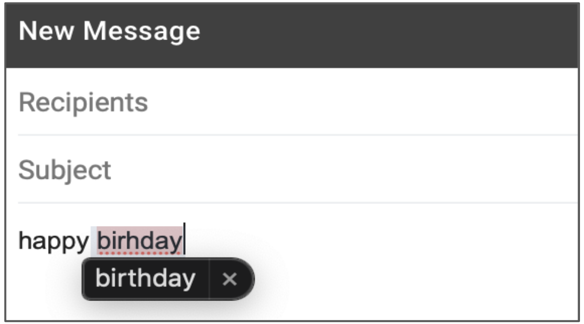
Happy birhday

Spelling and grammar

Change **birhday** to:

birthday

Ignore



New Message

Recipients

Subject

happy birhday

birthday ×

What is autocorrect?

- Example:

Happy birthday deah friend! 🎂

What is autocorrect ?

- Example:

✓

Happy birthday dear friend!

What is autocorrect?

- Example:

Happy birthday deer friend! 🦌 ??

How it works

1. Identify a misspelled word
2. Find strings n edit distance away
3. Filter candidates
4. Calculate word probabilities

How it works

1. Identify a misspelled word
2. Find strings n edit distance away
3. Filter candidates
4. Calculate word probabilities

deah

How it works

1. Identify a misspelled word
2. Find strings n edit distance away
3. Filter candidates
4. Calculate word probabilities

deah

_eah

d_ar

de_r

... *etc*

How it works

1. Identify a misspelled word
2. Find strings n edit distance away
3. Filter candidates
4. Calculate word probabilities

deah

yeah

dear

dean

... *etc*

How it works

1. Identify a misspelled word
2. Find strings n edit distance away
3. Filter candidates
4. Calculate word probabilities

deah

yeah

dear

dean

... etc

How it works

1. Identify a misspelled word
2. Find strings n edit distance away
3. Filter candidates
4. Calculate word probabilities

deah → dear ☑

yeah

dear

dean

... *etc*

deeplearning.ai

Building the model

Building the model

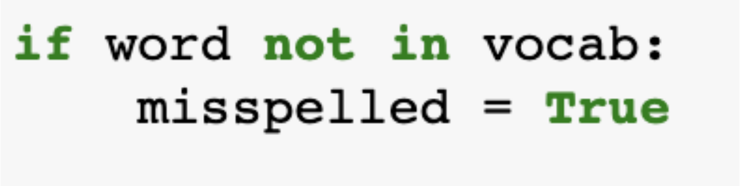
1. Identify a misspelled word
2. Find strings n edit distance away
3. Filter candidates
4. Calculate word probabilities

Building the model

1. Identify a misspelled word
2. Find strings n edit distance away
3. Filter candidates
4. Calculate word probabilities

Building the model

1. Identify a misspelled word

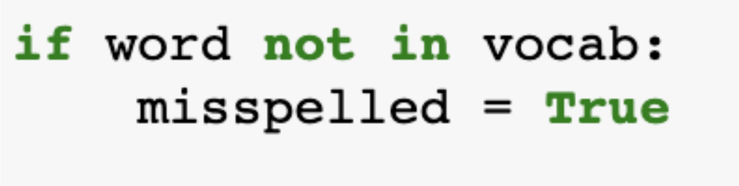


```
if word not in vocab:
    misspelled = True
```

deah ?? 🤔

Building the model

1. Identify a misspelled word

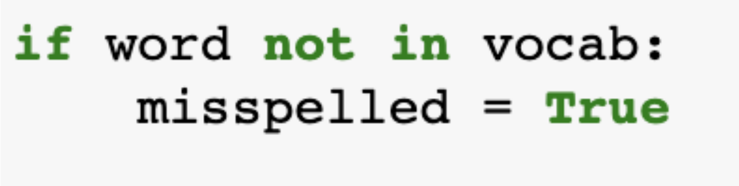


```
if word not in vocab:
    misspelled = True
```

deah ✗

Building the model

1. Identify a misspelled word



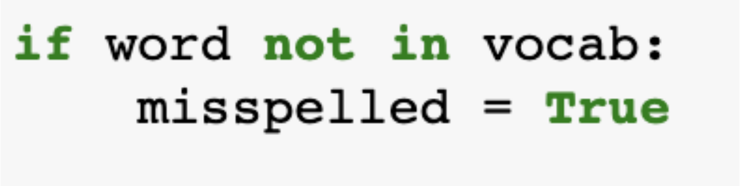
```
if word not in vocab:
    misspelled = True
```

deah ✗

deer 🦌 ☑

Building the model

1. Identify a misspelled word



```
if word not in vocab:
    misspelled = True
```

deah ✗

Happy birthday deer ! 🦌 🎂 ☑

Building the model

1. Identify a misspelled word
2. Find strings n edit distance away
3. Filter candidates
4. Calculate word probabilities

Building the model

2. Find strings n edit distance away

Building the model

2. Find strings n edit distance away

- Edit: an operation performed on a string to change it

Building the model

2. Find strings n edit distance away

- Edit: an operation performed on a string to change it
- Insert (add a letter)

Building the model

2. Find strings n edit distance away

- Edit: an operation performed on a string to change it

- Insert (add a letter)

  ‘to’: ‘top’, ‘two’ …

Building the model

2. Find strings n edit distance away

- Edit: an operation performed on a string to change it

- Insert (add a letter)

  ‘to’: ‘top’, ‘two’ …
- Delete (remove a letter)

Building the model

2. Find strings n edit distance away

- Edit: an operation performed on a string to change it

- Insert (add a letter)

  ‘to’: ‘top’, ‘two’ …

- Delete (remove a letter)

  ‘hat’: ‘ha’, ‘at’, ‘ht’

Building the model

2. Find strings n edit distance away

- Edit: an operation performed on a string to change it

- Insert (add a letter)
  ‘to’: ‘top’, ‘two’ …
- Delete (remove a letter)
  ‘hat’: ‘ha’, ‘at’, ‘ht’
- Switch (swap 2 adjacent letters)

Building the model

2. Find strings n edit distance away

- Edit: an operation performed on a string to change it

- Insert (add a letter)

  ‘to’: ‘top’, ‘two’ …
- Delete (remove a letter)

  ‘hat’: ‘ha’, ‘at’, ‘ht’
- Switch (swap 2 adjacent letters) ‘eta’: ‘eat’, ‘tea’

Building the model

2. Find strings n edit distance away

- Edit: an operation performed on a string to change it

- Insert (add a letter) 'to': 'top', 'two' ...
- Delete (remove a letter) 'hat': 'ha', 'at', 'ht'
- Switch (swap 2 adjacent letters) 'eta': 'eat', 'tea' 'ate'

✕

Building the model

2. Find strings n edit distance away

- Edit: an operation performed on a string to change it

- Insert (add a letter)

  ‘to’: ‘top’, ‘two’ …
- Delete (remove a letter)

  ‘hat’: ‘ha’, ‘at’, ‘ht’
- Switch (swap 2 adjacent letters) ‘eta’: ‘eat’, ‘tea’
- Replace (change 1 letter to another)

Building the model

2. Find strings n edit distance away

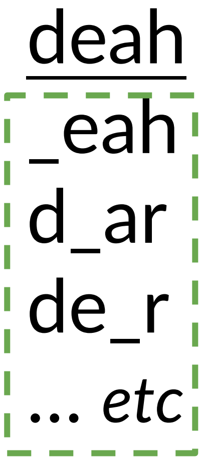
- Edit: an operation performed on a string to change it

- Insert (add a letter) 'to': 'top', 'two' ...
- Delete (remove a letter) 'hat': 'ha', 'at', 'ht'
- Switch (swap 2 adjacent letters) 'eta': 'eat', 'tea'
- Replace (change 1 letter to another) 'jaw': 'jar',

Building the model

2. Find strings n edit distance away

- Given a string find all possible strings that are n edit distance away using
    - Input
    - Delete
    - Switch
    - Replace



deah

_eah

d_ar

de_r

... *etc*

Building the model

1. Identify a misspelled word
2. Find strings n edit distance away
3. Filter candidates
4. Calculate word probabilities

Building the model

3. Filter candidates

deah

_eah

d_ar

de_r

... *etc*

Building the model

3. Filter candidates

deah deah

_eah yeah

d_ar → dear

de_r dean

… *etc* … *etc*

deeplearning.ai

Building the model II

Building the model

1. Identify a misspelled word
2. Find strings n edit distance away
3. Filter candidates
4. Calculate word probabilities

Building the model

4. Calculate word probabilities

Building the model

4. Calculate word probabilities

Example: “I am happy because I am learning”

Word	Count
I	2
am	2
happy	1
because	1
learning	1

Total : 7

Building the model

4. Calculate word probabilities

Example: “I am happy because I am learning”

Word	Count
I	2
am	2
happy	1
because	1
learning	1

Total : 7

Building the model

4. Calculate word probabilities

Example: “I am happy because I am learning”

Word	Count
I	2
am	2
happy	1
because	1
learning	1

Total : 7

Building the model

4. Calculate word probabilities

Example: “I am happy because I am learning”

Word	Count
I	2
am	2
happy	1
because	1
learning	1

Total : 7

Building the model

4. Calculate word probabilities

Example: “I am happy because I am learning”

$$P(w) = \frac{C(w)}{V}$$

$P(w)$ Probability of a word

$C(w)$ Number of times the word appears

V Total size of the corpus

Word	Count
I	2
am	2
happy	1
because	1
learning	1

Total : 7

Building the model

4. Calculate word probabilities

Example: “I am happy because I am learning”

$$P(w) = \frac{C(w)}{V}$$

$$\boxed{P(\mathrm{am}) = \frac{C(\mathrm{am})}{V} = \frac{2}{7}}$$

$P(w)$ Probability of a word

$C(w)$ Number of times the word appears

V Total size of the corpus

Word	Count
I	2
am	2
happy	1
because	1
learning	1

Total : 7

Building the model

4. Calculate word probabilities

deah

yeah

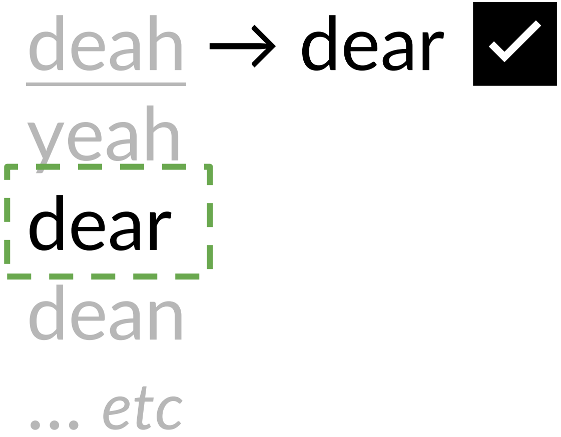
dear

dean

… *etc*

Building the model

4. Calculate word probabilities



deah → dear ☑

yeah

dear

dean

… *etc*

Summary

1. Identify a misspelled word
2. Find strings n edit distance away

   Insert
   Delete
   Switch
   Replace

1. Filter candidates
2. Calculate word probabilities

$$P(w) = \frac{C(w)}{V}$$

deah → dear ☑

yeah

dear

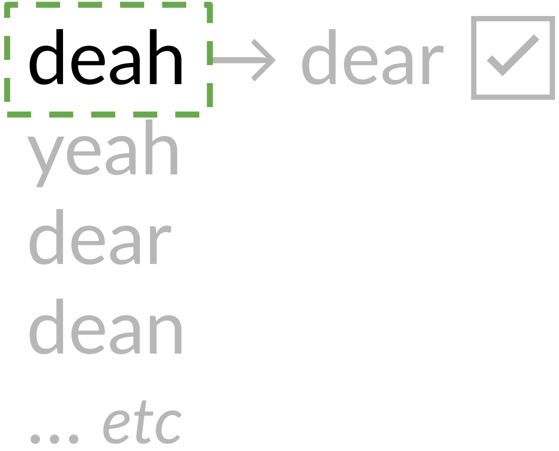
dean

… *etc*

Summary

1. Identify a misspelled word
2. Find strings n edit distance away
   - Insert
   - Delete
   - Switch
   - Replace
1. Filter candidates
2. Calculate word probabilities

$$P(w) = \frac{C(w)}{M}$$



deah → dear ☑

yeah

dear

dean

… *etc*

Summary

1. Identify a misspelled word
2. Find strings n edit distance away
   - Insert
   - Delete
   - Switch
   - Replace

1. Filter candidates

2. Calculate word probabilities

$$P(w) = \frac{C(w)}{M}$$

deah → dear ☑

yeah

dear

dean

… *etc*

Summary

1. Identify a misspelled word
2. Find strings n edit distance away
   - Insert
   - Delete
   - Switch
   - Replace
1. Filter candidates
2. Calculate word probabilities

$$P(w) = \frac{C(w)}{M}$$

deah → dear ☑

_eah

d_ar

de_r

... *etc*

Summary

1. Identify a misspelled word
2. Find strings n edit distance away

    Insert
    Delete
    Switch
    Replace

1. Filter candidates
2. Calculate word probabilities

$$P(w) = \frac{C(w)}{M}$$

<u>deah</u> → dear ☑

yeah

dear

dean

… *etc*

Summary

1. Identify a misspelled word
2. Find strings n edit distance away
    - Insert
    - Delete
    - Switch
    - Replace
1. Filter candidates
2. Calculate word probabilities

$$P(w) = \frac{C(w)}{M}$$

deah → dear ☑

yeah

dear

dean

... *etc*

Summary

1. Identify a misspelled word
2. Find strings n edit distance away

   Insert
   Delete
   Switch
   Replace

1. Filter candidates
2. Calculate word probabilities

$$P(w) = \frac{C(w)}{M}$$

deah → dear ✓

yeah

dear

dean

... etc

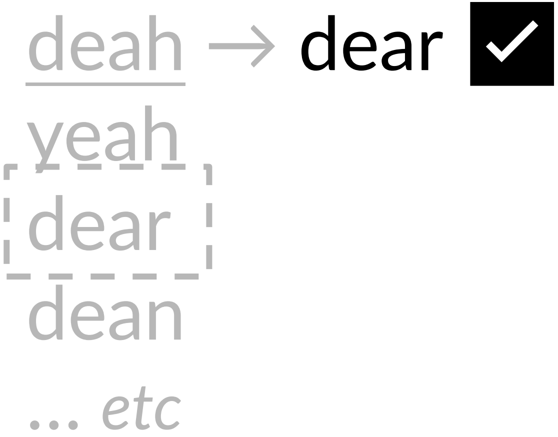
Summary

1. Identify a misspelled word
2. Find strings n edit distance away

    Insert
    Delete
    Switch
    Replace

1. Filter candidates
2. Calculate word probabilities

$$P(w) = \frac{C(w)}{M}$$



deah → dear ✓

yeah

dear

dean

… *etc*

deeplearning.ai

Minimum edit distance

Minimum edit distance

- How to evaluate similarity between 2 strings?

- Minimum number of edits needed to transform 1 string into the other

- Spelling correction, document similarity, machine translation, DNA sequencing, and more

Minimum edit distance

- Edits:

- Insert (add a letter) ‘to’: ‘top’, ‘two’ …
- Delete (remove a letter) ‘hat’: ‘ha’, ‘at’, ‘ht’
- Replace (change 1 letter to another) ‘jaw’: ‘jar’, ‘paw’, …

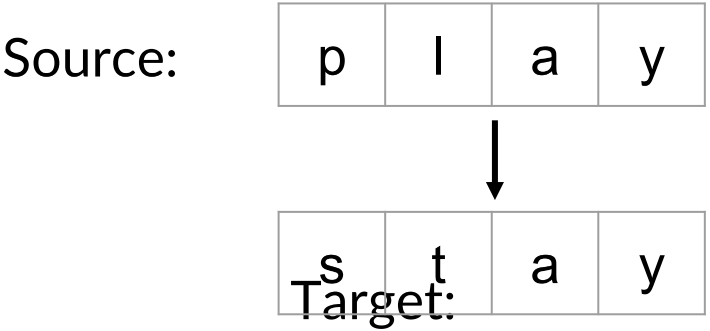
Minimum edit distance

- Example:

Source:	p	l	a	y

Minimum edit distance

- Example:



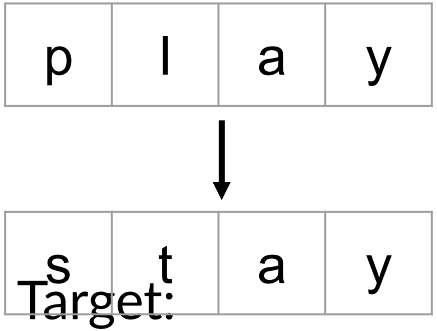
Source: p l a y

s t a y

Target:

Minimum edit distance

- Example:



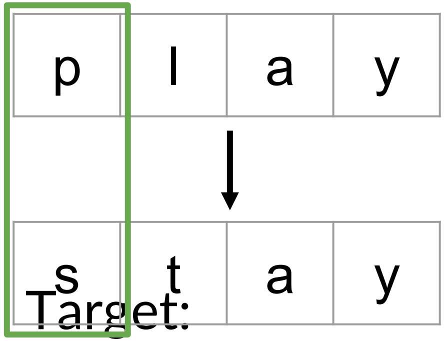
Source:	p	l	a	y
Target:	s	t	a	y

What is the minimum number of edits to make this happen ?

Minimum edit distance

- Example:

Source:



p	l	a	y

s	t	a	y

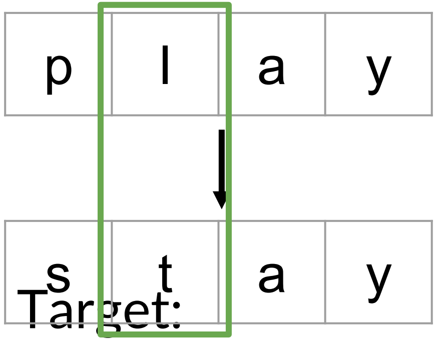
Target:

p → s : replace

Minimum edit distance

- Example:

Source:



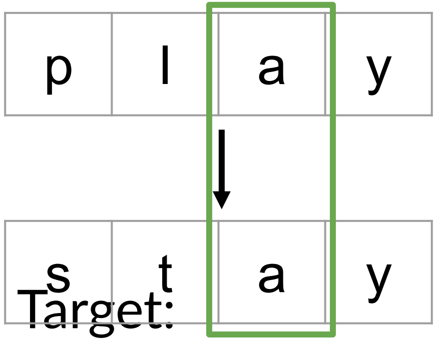
p	l	a	y
s	t	a	y

Target:

p → s : replace
l → t : replace

Minimum edit distance

- Example:

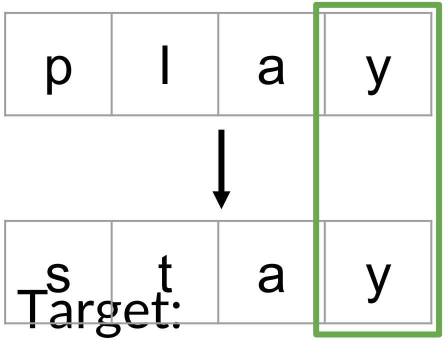


Source:	p	l	a	y
Target:	s	t	a	y

p → s : replace
l → t : replace

Minimum edit distance

- Example:

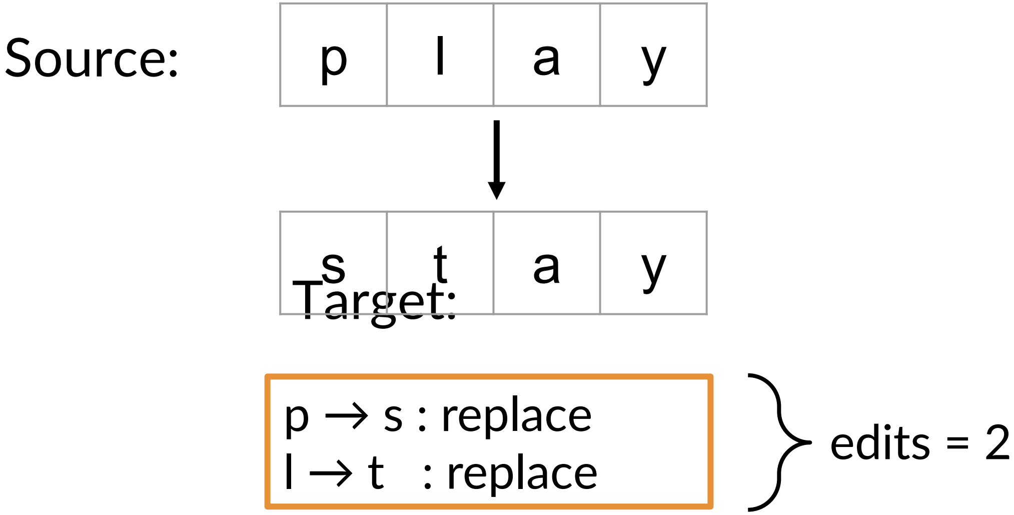


Source:	p	l	a	y
Target:	s	t	a	y

p → s : replace
l → t : replace

Minimum edit distance

- Example:



Source:

p	l	a	y

s	t	a	y

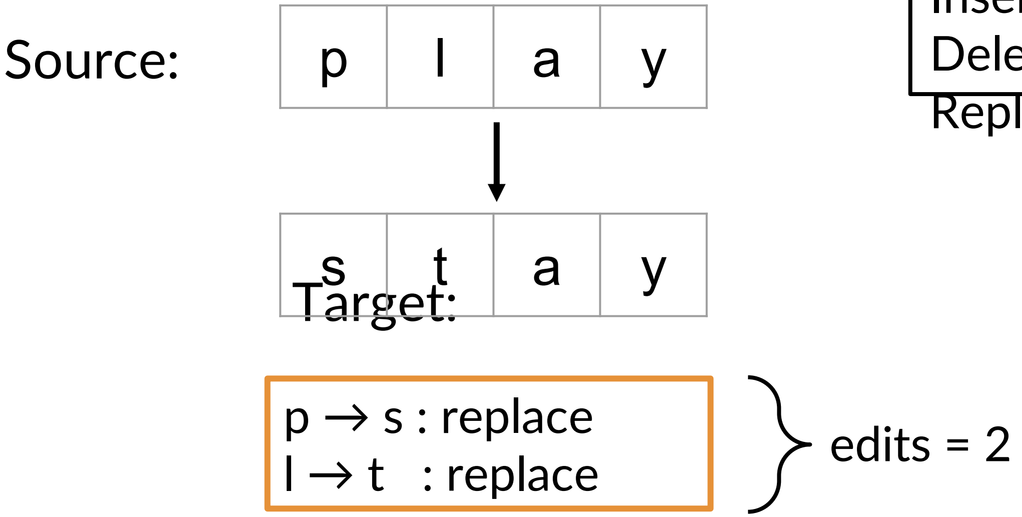
Target:

p → s : replace
l → t : replace

edits = 2

Minimum edit distance

- Example:



Source: p l a y

s t a y
Target:

p → s : replace
l → t : replace

edits = 2

Edit cost:

Insert	1
Delete	1
Replace	2

Minimum edit distance

- Example:

Source:	p	l	a	y
	↓			
Target:	s	t	a	y

p → s : replace
l → t : replace

edits = 2

Edit cost:

Insert	1
Delete	1
Replace	2

edit distance = 2 * 2 = 4

Minimum edit distance

- Example:

c	o	n	v	o	l	u	t	i	o	n	a	l	n	e	u	r	a	l	n	e	t	w	o	r	k

Minimum edit distance

- Example:

c	o	n	v	o	l	u	t	i	o	n	a	l	n	e	u	r	a	l	n	e	t	w	o	r	k

CCAAGGGGTGACTCTAGTTTAATATAACTGAGATCAAATTATATGGGTGAT⍰ !!

Minimum edit distance algorithm

deeplearning.ai

Minimum edit distance

Source: play → Target: stay

		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

Minimum edit distance

Source: play → Target: stay

		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

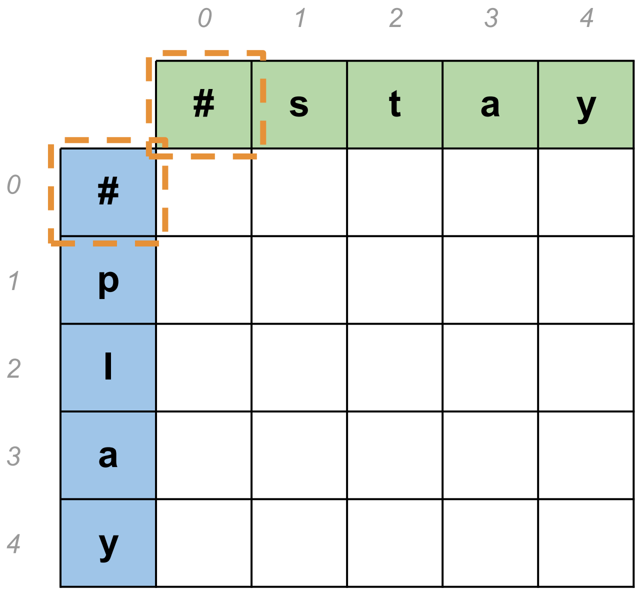
Minimum edit distance

Source: play → Target: stay

		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

Minimum edit distance

Source: play → Target: stay



		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

Minimum edit distance

Source: play → Target: stay

		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

Minimum edit distance

Source: play → Target: stay

D[ ]

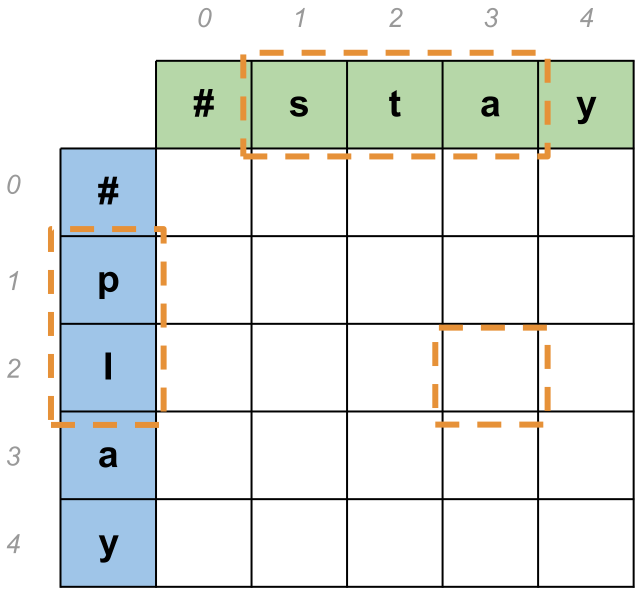
		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

Minimum edit distance

Source: play → Target: stay

$D[\ ]$

$D[2,3]$ = pl → sta



		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

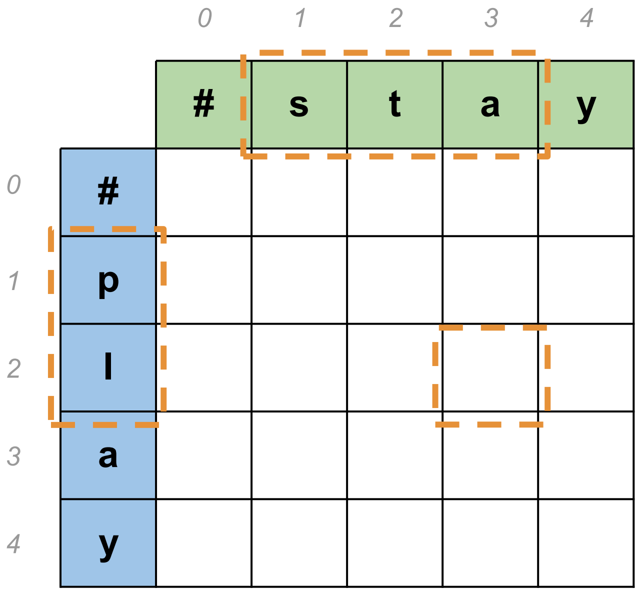
Minimum edit distance

Source: play → Target: stay

D[ ]

D[2,3] = pl → sta

D[2,3] = source[:2] → target[:3]



		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

Minimum edit distance

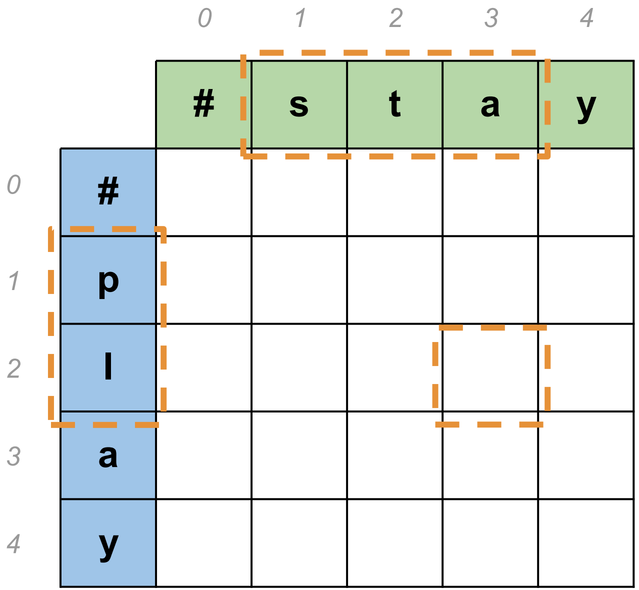
Source: play → Target: stay

D[ ]

D[2,3] = pl → sta

D[2,3] = source[:2] → target[:3]

D[*i*, *j*] = source[: *i*] → target[: *j*]



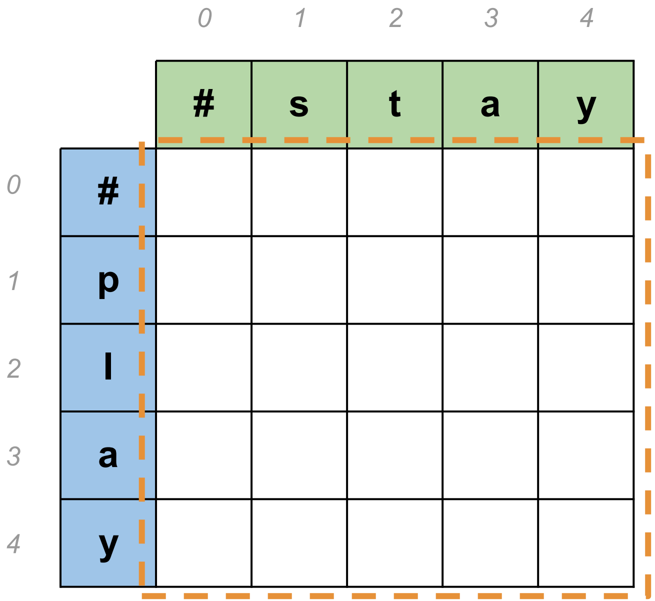
		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

Minimum edit distance

Source: play → Target: stay

$D[\ ]$

$D[\,i, j\,] =$ source[: i] → target[: j]



		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

Minimum edit distance

Source: play → Target: stay

$D[\ ]$

$D[\, i, j\,] =$ source[: i] → target[: j]

$D[\, m, n\,] =$ source → target

		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

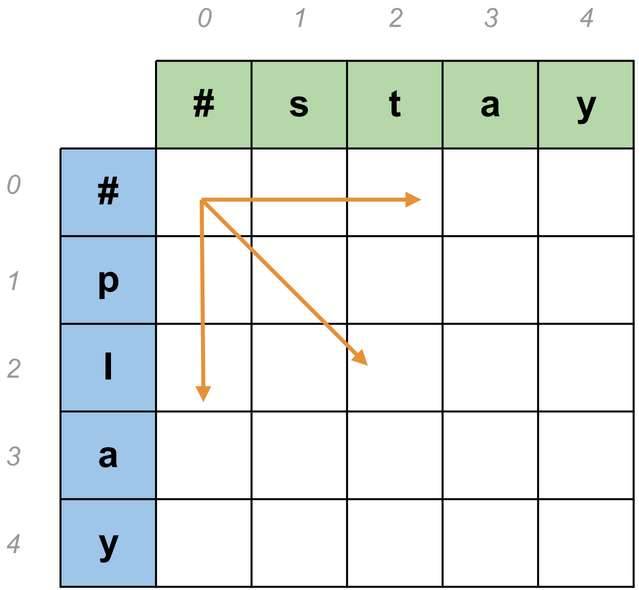
Minimum edit distance

Source: play → Target: stay

$D[\,]$

$D[\,i, j\,] =$ source[: i] → target[: j]

$D[\,m, n\,] =$ source → target



		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

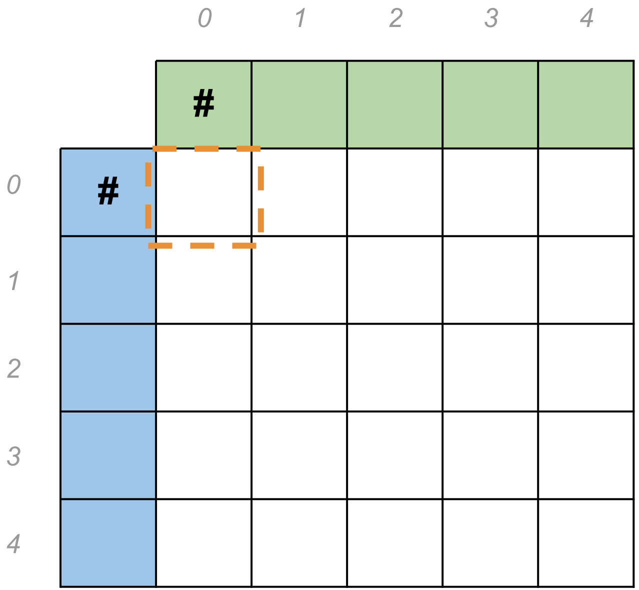
		0	1	2	3	4
		#	s	t	a	y
0	#					
1	p					
2	l					
3	a					
4	y					

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

→



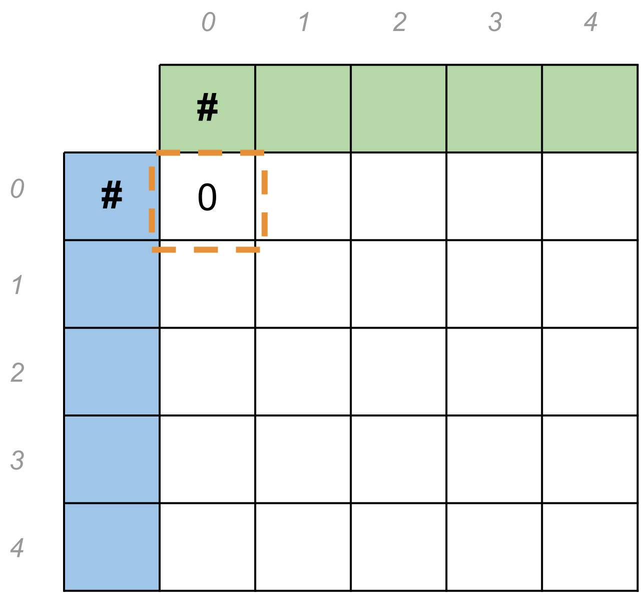
		0	1	2	3	4
		#				
0	#					
1						
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

→



		0	1	2	3	4
		#				
0	#	0				
1						
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → #

		0	1	2	3	4
		#				
0	#	0				
1	p					
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → #

delete

		0	1	2	3	4
		#				
0	**#**	0				
1	**p**	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

→ s

		0	1	2	3	4
		#	s			
0	#	0				
1	p	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

→ s

insert

		0	1	2	3	4
		#	**s**			
0	**#**	0	1			
1	**p**	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

		0	1	2	3	4
		#	s			
0	#	0	1			
1	p	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

insert + *delete*: p → ps → s

		0	1	2	3	4
		#	**s**			
0	**#**	0	1			
1	**p**	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

insert + *delete*: p → ps → s

		0	1	2	3	4
		#	**s**			
0	**#**	0	1			
1	**p**	1				
2						
3						
4						

Minimum edit distance

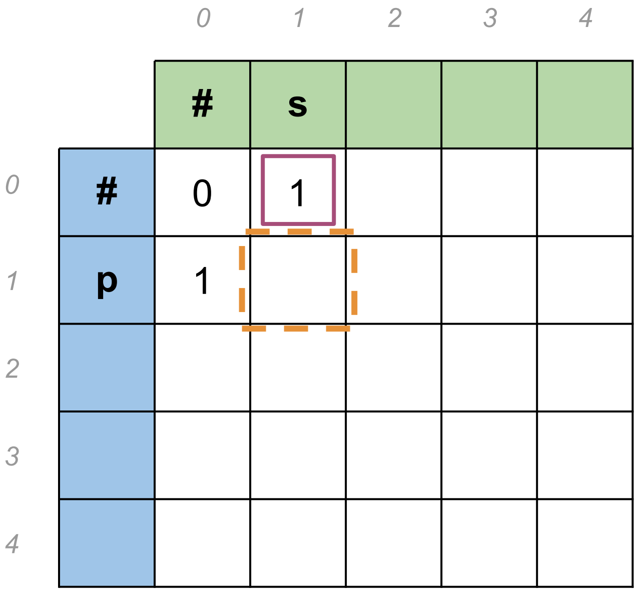
Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

insert + *delete*: p → ps → s:

2



		0	1	2	3	4
		#	s			
0	#	0	1			
1	p	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

insert + *delete*: p → ps → s:

2

delete + *insert*: p → # → s

		0	1	2	3	4
		#	s			
0	#	0	1			
1	p	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

insert + *delete*: p → ps → s:

2

delete + *insert*: p → # → s

		0	1	2	3	4
		#	**s**			
0	**#**	0	1			
1	**p**	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

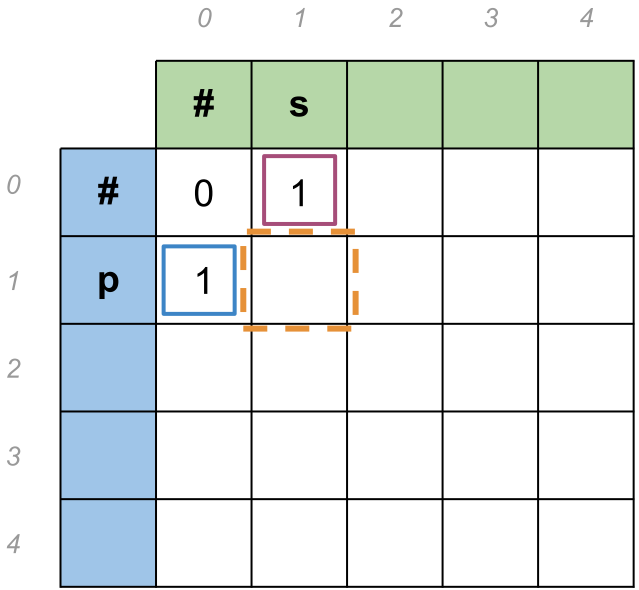
Cost: insert: 1, delete: 1, replace: 2

p → s

insert + *delete*: p → ps → s:

2

delete + *insert*: p → # → s: 2



		0	1	2	3	4
		#	s			
0	#	0	1			
1	p	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

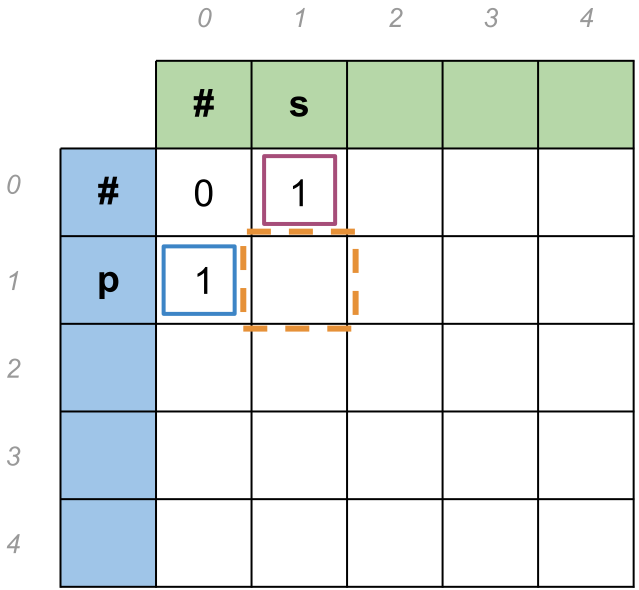
Cost: insert: 1, delete: 1, replace: 2

p → s

insert + *delete*: p → ps → s: 2

delete + *insert*: p → # → s: 2

replace: p → s



		0	1	2	3	4
		#	**s**			
0	**#**	0	1			
1	**p**	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

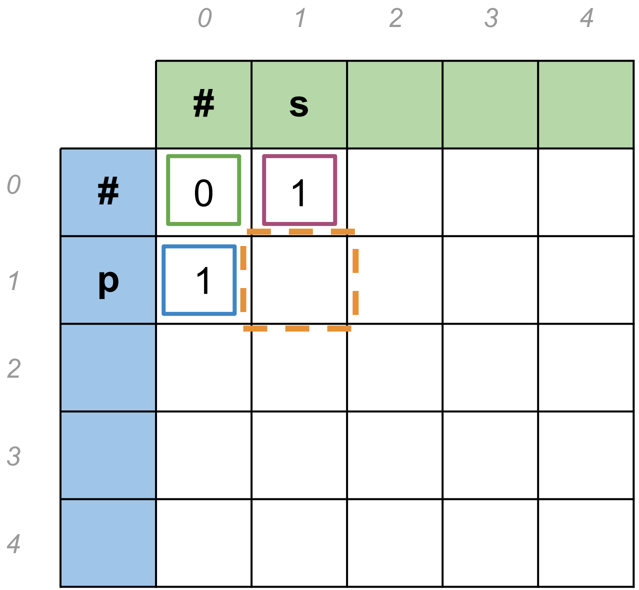
p → s

insert + *delete*: p → ps → s:

2

delete + *insert*: p → # → s: 2

replace: p → s



		0	1	2	3	4
		#	**s**			
0	**#**	0	1			
1	**p**	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

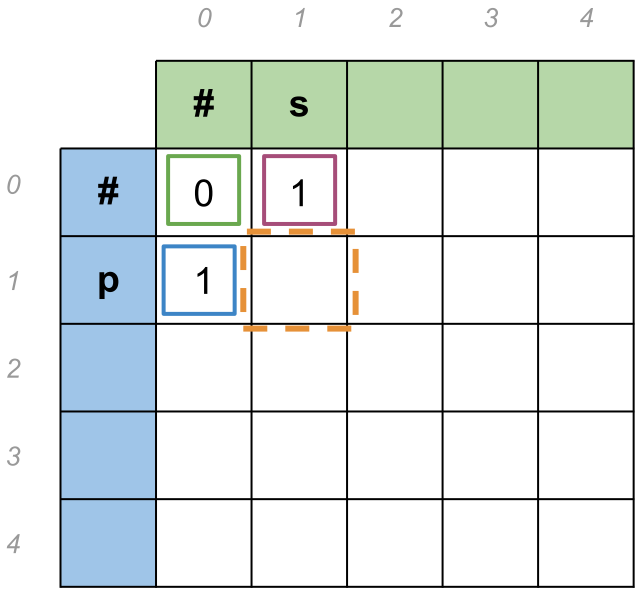
Cost: insert: 1, delete: 1, replace: 2

p → s

insert + *delete*: p → ps → s: 2

delete + *insert*: p → # → s: 2

replace: p → s: 2



		0	1	2	3	4
		#	**s**			
0	**#**	0	1			
1	**p**	1				
2						
3						
4						

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

insert + *delete*: p → ps → s:
2

delete + *insert*: p → # → s: 2

replace: p → s:
2

		0	1	2	3	4
		#	**s**			
0	**#**	0	1			
1	**p**	1	2			
2						
3						
4						

deeplearning.ai

Minimum edit distance algorithm II

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

		0	1	2	3	4
		#	s	t	a	y
0	#	0	1			
1	p	1	2			
2	l					
3	a					
4	y					

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

play → #

		0	1	2	3	4
		#	s	t	a	y
0	#	0	1			
1	p	1	2			
2	l					
3	a					
4	y					

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

play → #

$$D[\,i, j\,] = D[\,i\text{-}1, j\,] + del\_cost$$

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1			
1	**p**	1	2			
2	**l**					
3	**a**					
4	**y**					

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

play → #

$$D[\ i, j\ ] = D[\ i\text{-}1, j\ ] + del\_cost$$

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1			
1	**p**	1	2			
2	**l**	2				
3	**a**	3				
4	**y**	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

play → #

$D[\,i, j\,] = D[\,i\text{-}1, j\,] + \mathit{del\_cost}$

$D[4,0]$ = play → #

= source[:4] → target[0]

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1			
1	**p**	1	2			
2	**l**	2				
3	**a**	3				
4	**y**	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

# → play

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1			
1	**p**	1	2			
2	**l**	2				
3	**a**	3				
4	**y**	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

# → play

$$D[\ i, j\ ] = D[\ i, j\text{-}1\ ] + ins\_cost$$

		0	1	2	3	4
		#	s	t	a	y
0	#	0	1			
1	p	1	2			
2	l	2				
3	a	3				
4	y	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

# → play

$D[\ i, j\ ] = D[\ i, j\text{-}1\ ] + \mathit{ins\_cost}$

		0	1	2	3	4
		#	s	t	a	y
0	#	0	1	2	3	4
1	p	1	2			
2	l	2				
3	a	3				
4	y	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

D[*i*, *j*] =

min { *D*[*i* - 1, *j*] + *del_cost*
D[*i*, *j* - 1] + *ins_cost*
D[*i* - 1, *j* - 1] + { *rep_cost*;   if *src*[*i*] ≠
tar[*j*]
0;   if *src*[*i*]
= *tar*[*j*]

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2			
2	**l**	2				
3	**a**	3				
4	**y**	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

$$D[i, j] = min \begin{cases} \boxed{D[i - 1, j] + del\_cost} \\ D[i, j - 1] + ins\_cost \\ D[i - 1, j - 1] + \begin{cases} rep\_cost; & \text{if } src[i] \neq tar[j] \\ 0; & \text{if } src[i] = tar[j] \end{cases} \end{cases}$$

		0	1	2	3	4
		#	s	t	a	y
0	#	0	1	2	3	4
1	p	1	2			
2	l	2				
3	a	3				
4	y	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

D[*i*, *j*] =

D[*i* - 1, *j*] + *del_cost*

min *D*[*i*, *j* - 1] + *ins_cost*

D[*i* - 1, *j* - 1] + *rep_cost*; if *src*[*i*] ≠

tar[*j*]

0; if *src*[*i*]

= *tar*[*j*]

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2			
2	**l**	2				
3	**a**	3				
4	**y**	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

$$D[i, j] = \min\begin{cases} D[i - 1, j] + del\_cost \\ D[i, j - 1] + ins\_cost \\ D[i - 1, j - 1] + \begin{cases} rep\_cost; & \text{if } src[i] \neq tar[j] \\ 0; & \text{if } src[i] = tar[j] \end{cases} \end{cases}$$

		0	1	2	3	4
		#	s	t	a	y
0	#	0	1	2	3	4
1	p	1	2			
2	l	2				
3	a	3				
4	y	4				

Minimum edit distance

Source: play → Target: stay

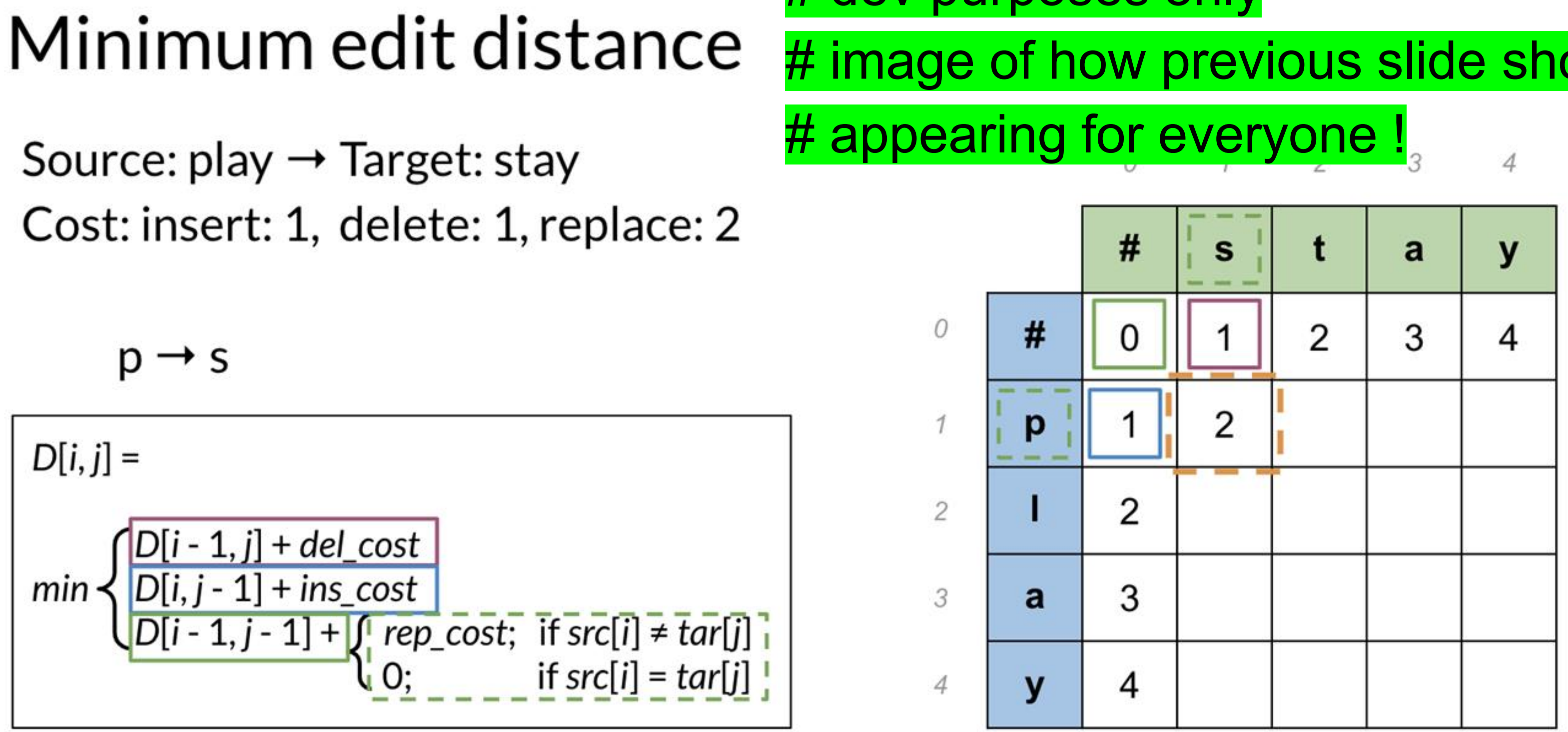
Cost: insert: 1, delete: 1, replace: 2

p → s

$$D[i, j] = \min \begin{cases} D[i - 1, j] + del\_cost \\ D[i, j - 1] + ins\_cost \\ D[i - 1, j - 1] + \begin{cases} rep\_cost; & \text{if } src[i] \neq tar[j] \\ 0; & \text{if } src[i] = tar[j] \end{cases} \end{cases}$$

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2			
2	**l**	2				
3	**a**	3				
4	**y**	4				

# dev purposes only
# image of how previous slide should be
# appearing for everyone !



Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

$$D[i, j] = \min \begin{cases} \boxed{D[i-1, j] + del\_cost} \\ \boxed{D[i, j-1] + ins\_cost} \\ \boxed{D[i-1, j-1]} + \boxed{\begin{cases} rep\_cost; & \text{if } src[i] \neq tar[j] \\ 0; & \text{if } src[i] = tar[j] \end{cases}} \end{cases}$$

		0	*1*	*2*	*3*	*4*
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2			
2	**l**	2				
3	**a**	3				
4	**y**	4				

Don't include text or images below this line. Delete this text and red line in the master template once you're finished with your slide creation

Minimum edit distance

$$D[i, j] = min\begin{cases} D[i - 1, j] + del\_cost \\ D[i, j - 1] + ins\_cost \\ D[i - 1, j - 1] + \begin{cases} rep\_cost; & \text{if } src[i] \neq tar[j] \\ 0; & \text{if } src[i] = tar[j] \end{cases} \end{cases}$$

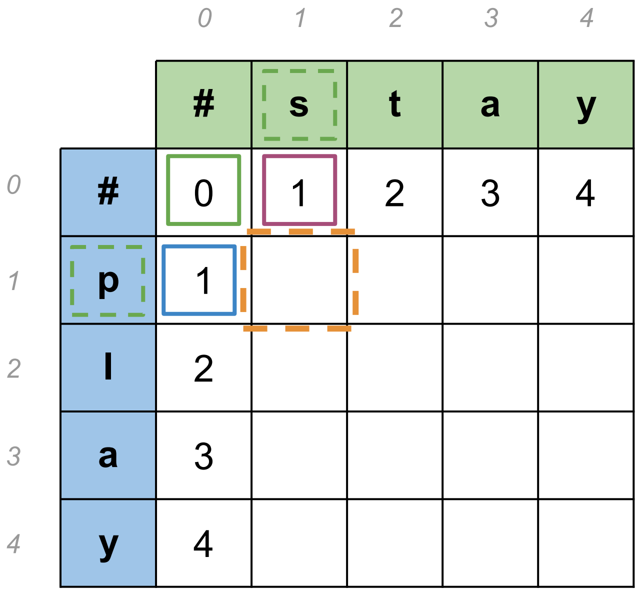
Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

FORMULAS BUILDING ONLY

EQUATION USED IN NEXT SLIDES

$$D[i, j] = min\begin{cases} D[i - 1, j] + del\_cost \\ D[i, j - 1] + ins\_cost \\ D[i - 1, j - 1] + \begin{cases} rep\_cost; & \text{if } src[i] \neq tar[j] \\ 0; & \text{if } src[i] = tar[j] \end{cases} \end{cases}$$



		0	1	2	3	4
		#	s	t	a	y
0	#	0	1	2	3	4
1	p	1				
2	l	2				
3	a	3				
4	y	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

$D[\ i\text{-}1, j\ ] + 1 = 2$

$D[\ i, j\text{-}1\ ] + 1 = 2$

$D[\ i\text{-}1, j\text{-}1\ ] + 2 = 2$

		0	1	2	3	4
		#	s	t	a	y
0	#	0	1	2	3	4
1	p	1	2			
2	l	2				
3	a	3				
4	y	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

$$\boxed{D[\,i\text{-}1, j\,] + 1 = 2}$$

$$D[\,i, j\text{-}1\,] + 1 = 2$$

$$D[\,i\text{-}1, j\text{-}1\,] + 2 = 2$$

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	$\boxed{1}$	2	3	4
1	**p**	1	2			
2	**l**	2				
3	**a**	3				
4	**y**	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

$D[\, i\text{-}1, j \,] + 1 = 2$

$\boxed{D[\, i, j\text{-}1 \,] + 1 = 2}$

$D[\, i\text{-}1, j\text{-}1 \,] + 2 = 2$

		0	1	2	3	4
		#	s	t	a	y
0	#	0	1	2	3	4
1	p	1	2			
2	l	2				
3	a	3				
4	y	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

D[*i*-1, *j*] + 1 = 2

D[*i*, *j*-1] + 1 = 2

D[*i*-1, *j*-1] + 2 = 2

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2			
2	**l**	2				
3	**a**	3				
4	**y**	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

p → s

$$= 2 \quad \left. \begin{array}{l} D[\,i\text{-}1, j\,] + 1 = 2 \\ D[\,i, j\text{-}1\,] + 1 = 2 \\ D[\,i\text{-}1, j\text{-}1\,] + 2 = 2 \end{array} \right\} \min$$

		0	1	2	3	4
		#	s	t	a	y
0	#	0	1	2	3	4
1	p	1	2			
2	l	2				
3	a	3				
4	y	4				

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

		0	1	2	3	4
		#	s	t	a	y
0	#	0	1	2	3	4
1	p	1	2			
2	l	2				
3	a	3				
4	y	4				

Minimum edit distance

$$D[i,j] = min \begin{cases} D[i-1,j] + del\_cost \\ D[i,j-1] + ins\_cost \\ D[i-1,j-1] + \begin{cases} rep\_cost; & \text{if } src[i] \neq tar[j] \\ 0; & \text{if } src[i] = tar[j] \end{cases} \end{cases}$$

Source: to → Target: go

Cost: insert: 1, delete: 1, replace: 2

FOR QUIZ SETUP ONLY

... USED FOR IMAGES

ON QUIZ IN NEXT SLIDE

		0	1	2
		#	g	o
0	#	0	1	2
1	t	1	2	3
2	o	2	3	

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2	3	4	5
2	**l**	2	3	4	5	6
3	**a**	3	4	5	4	5
4	**y**	4	5	6	5	4

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

play → stay

D[*m*, *n*] = 4

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2	3	4	5
2	**l**	2	3	4	5	6
3	**a**	3	4	5	4	5
4	**y**	4	5	6	5	4

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

play → stay

D[*m*, *n*] = 4

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2	3	4	5
2	**l**	2	3	4	5	6
3	**a**	3	4	5	4	5
4	**y**	4	5	6	5	4

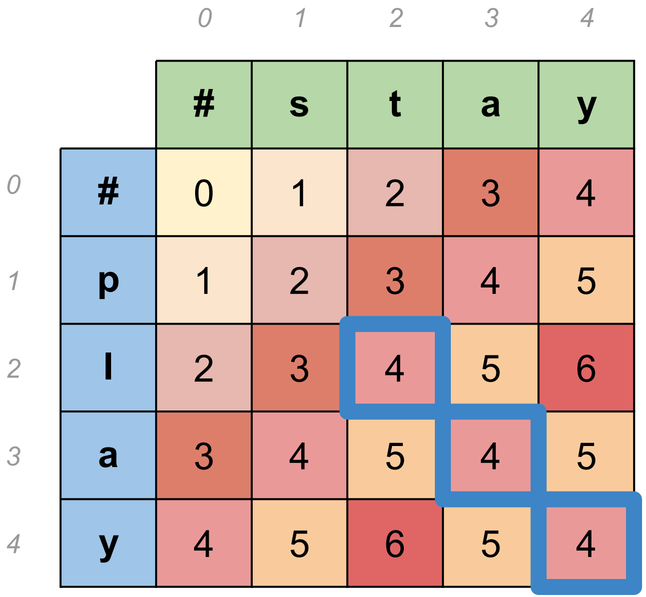
Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

play → stay

D[*m*, *n*] = 4



		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2	3	4	5
2	**l**	2	3	4	5	6
3	**a**	3	4	5	4	5
4	**y**	4	5	6	5	4

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

		0	1	2	3	4
		#	s	t	a	y
0	#	0	1	2	3	4
1	p	1	2	3	4	5
2	l	2	3	4	5	6
3	a	3	4	5	4	5
4	y	4	5	6	5	4

deeplearning.ai

Minimum edit distance algorithm III

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

- Levenshtein distance

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2	3	4	5
2	**l**	2	3	4	5	6
3	**a**	3	4	5	4	5
4	**y**	4	5	6	5	4

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

- Levenshtein distance
- Backtrace

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2	3	4	5
2	**l**	2	3	4	5	6
3	**a**	3	4	5	4	5
4	**y**	4	5	6	5	4

Minimum edit distance

Source: play → Target: stay

Cost: insert: 1, delete: 1, replace: 2

- Levenshtein distance
- Backtrace
- Dynamic programming

		0	1	2	3	4
		#	**s**	**t**	**a**	**y**
0	**#**	0	1	2	3	4
1	**p**	1	2	3	4	5
2	**l**	2	3	4	5	6
3	**a**	3	4	5	4	5
4	**y**	4	5	6	5	4

deeplearning.ai

Summary

Summary - learning objectives

- What is autocorrect ?
- Building the model
- Minimum edit distance
- Minimum edit distance algorithm

deah → dear ✅

yeah

dear

dean

... *etc*

	#	s	t	a	y
#	0	1	2	3	4
p	1	2	3	4	5
l	2	3	4	5	6
a	3	4	5	4	5
y	4	5	6	5	4