การแปลข้อมูลภาษา (Text-String) Vector (เป็นตัวเลข)

NLP : natural language processing

1) **วิธี** Bag of Words
- คำแต่ละคำเป็นอิสระต่อกัน (Independen)
- 1 ประโยค ได้ 1 Vector

EX 1. ไป หา อะไร กิน กัน 2. กิน ข้าว หรือ ยัง

1.1) **ตัดคำ** (Corpus) --> N-grams = 1

1. | ไป | หา | อะไร | กิน | กัน |
2. | กิน | ข้าว | หรือ | ยัง |

N-grams = 2

| ไปหา | อะไรกิน | กินกัน |

1.2) **สร้าง** Unique works **ของข้อมูล (คัดเอาตัวเหมือนกัน)** { } U { }

ไป, หา, อะไร, กิน, กัน, ข้าว, หรือ, ยัง

1.3) แปลง Corpus ให้เป็น Matrix (ปรากฏกี่ครั้งในข้อความ แต่ละประโยค)

	ไป	หา	อะไร	กัน	กิน	ข้าว	หรือ	ยัง
	1	1	1	1	1	0	0	0
	0	0	0	0	1	1	1	1
≶	1	1	1	1	2	1	1	1

1.4) Normailztion $\frac{\text{ค่าแต่ละ Row ใน Column นั้น}}{\text{≶ ของแต่ Colmn นั้น}}$

ไป	หา	อะไร	กิน	กัน	ข้าว	หรือ	ยัง
1	1	1	1	0.5	0	0	0
0	0	0	0	0.5	1	1	1

ข้อจำกัด

- คำแต่ละคำ ต้องเป็นอิสระต่อกัน (Independen)
  (แต่บางคำมีความสัมพันธ์ กับคำข้างๆ เช่น
  ฉันรักเธอ/ เธอรักกัน)

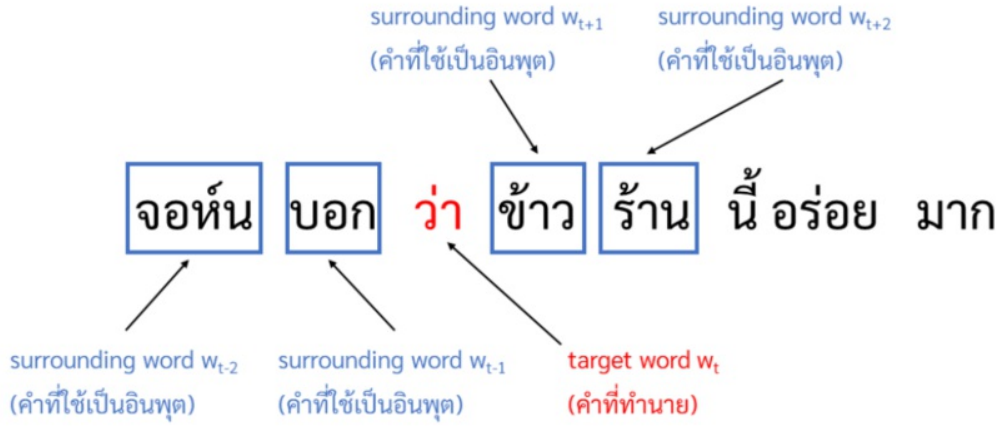
- 1 ประโยค ได้ 1 Vector
  (แต่บางครั้งอยากได้ 1 คำ 1 vector)

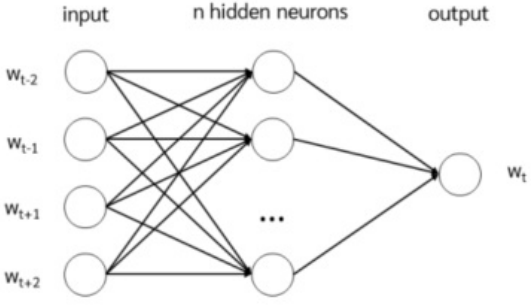
2) **วิธี** Words 2 Vec - representation

2.1) Continuous Bag-of-words (CBOW)

ดูคำรอบตัว ขึ้นอยู่กับที่เรากำหนด
แนะนำให้เป็นเลจคี่ เพื่อจะได้คำกลาง

CBOW ใช้ context หรือ surrounding words เป็น input ตัวอย่างถ้าให้ C (context window)
กำหนดให้ C=5 จะได้ input เป็นคำในตำแหน่งที่ w_{t-2} , w_{t-1} , w_{t+1} และ w_{t+2} จากนั้นทำนาย target word



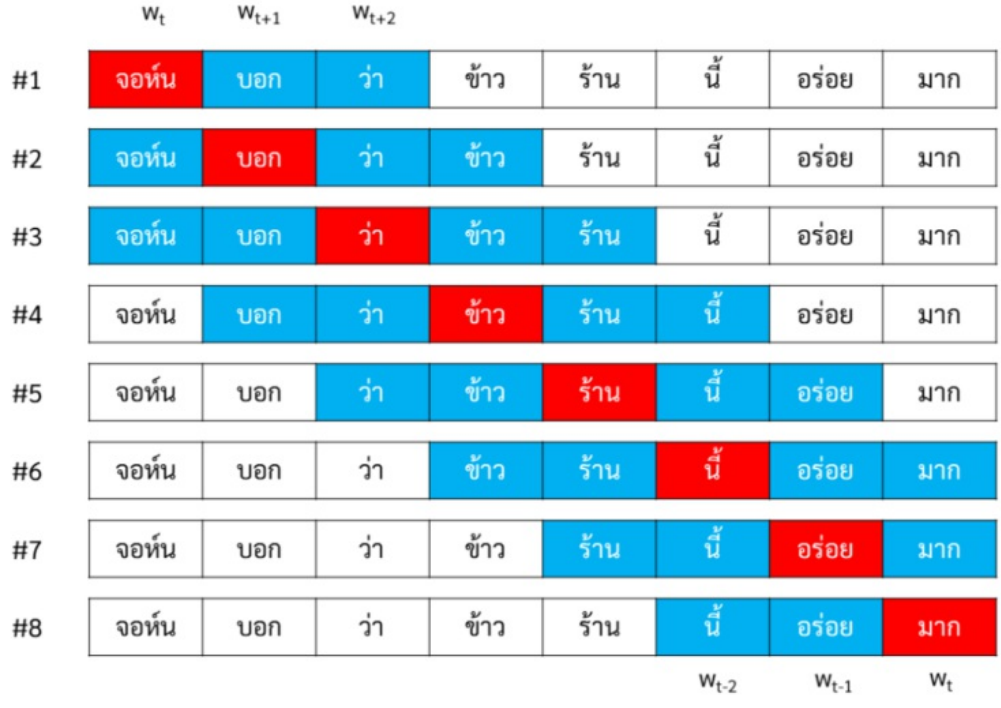


- ให้ Wt เป็น Target word (Center Word)
- ให้ Wt-2, Wt-1, Wt+1, Wt+2 เป็น Context (Surroundion words) คำที่อยู่ข้างๆ กี่คำก็ได้ตามกำหนด

EX

1. ไป หา อะไร กิน กัน
2. กิน ข้าว หรือ ยัง
3. ทำไม ไม่ กิน ข้าว
4. จอห์น กำลัง กิน ข้าว
5. ฉัน ยัง ไม่ หิว

1) ทำ sliding window
C = 5



2) หา Unique words เอาเฉพาะคำที่ไม่เหมือนกัน
-> ได้ Feature, Target จาก C = 5

Feature			
บอก	ว่า		
จอห์น	ว่า	ข้าว	
จอห์น	บอก	ข้าว	ร้าน
บอก	ว่า	ร้าน	นี้
ว่า	ข้าว	นี้	อร่อย
ข้าว	ร้าน	อร่อย	มาก
ร้าน	นี้	มาก	
นี้	อร่อย		

Target
จอห์น
บอก
ว่า
ข้าว
ร้าน
นี้
อร่อย
มาก

3) ทำเป็น one hot encoding ทั้ง Feature และ Target

Feature							
ข้าว	จอห์น	นี้	บอก	มาก	ร้าน	ว่า	อร่อย
0	0	0	1	0	0	1	0
1	1	0	0	0	0	1	0
1	1	0	1	0	1	0	0
0	0	1	1	0	1	1	0
1	0	1	0	0	0	1	1
1	0	0	0	1	1	0	1
0	0	1	0	1	1	0	0
0	0	1	0	0	0	0	1

Target							
ข้าว	จอห์น	นี้	บอก	มาก	ร้าน	ว่า	อร่อย
0	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	1
0	0	0	0	1	0	0	0

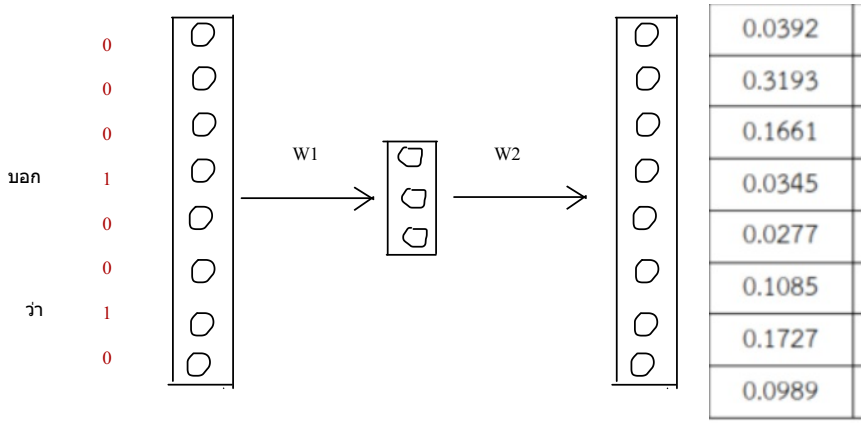
4) เมื่อได้ feature และ target matrices มาแล้ว ก็นำไปเทรนด้วย Neural network หรือ ML อื่นๆ ก็ได้ (1 hidden lavers, 3 nodes) หลังจากเทรนไป 1000 iterations ได้ผลลัพธ์ดังนี้

	ข้าว	จอห์น	นี้	บอก	มาก	ร้าน	ว่า	อร่อย
จอห์น	0.0392	0.0961	0.2108	0.0479	0.0429	0.1844	0.2653	0.1133
บอก	0.3193	0.2013	0.0806	0.0063	0.1769	0.0909	0.1035	0.0212
ว่า	0.1661	0.0184	0.1168	0.0032	0.1944	0.1512	0.0127	0.3371
ข้าว	0.0345	0.1903	0.0623	0.112	0.2348	0.0965	0.0482	0.2214
ร้าน	0.0277	0.2135	0.1863	0.007	0.2748	0.0044	0.0723	0.214
นี้	0.1085	0.074	0.1391	0.0612	0.0832	0.0963	0.4125	0.0252
อร่อย	0.1727	0.2398	0.3143	0.0664	0.0015	0.1396	0.0034	0.0623
มาก	0.0989	0.1725	0.0412	0.0753	0.1354	0.285	0.0084	0.1833

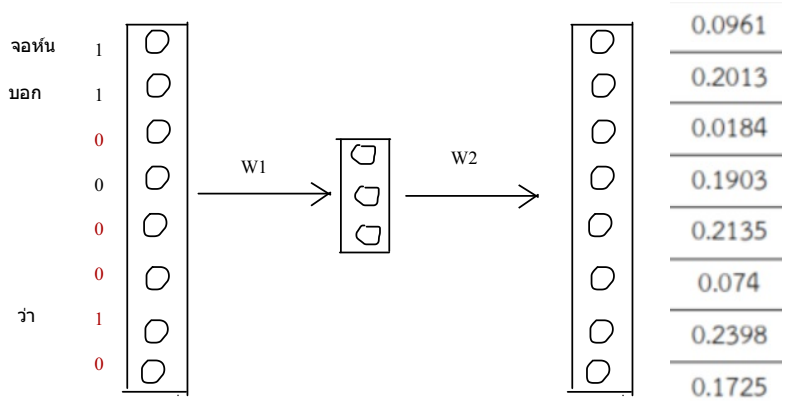
Vecter "ข้าว"

Vecter "บอก"

Vecter "ข้าว"



Vecter "จอห์น"



2.2) Skip-gram

ให้ w_t เป็น target word หรือ center word

ให้ w_{t-2} , w_{t-1} , w_{t+1} , w_{t+2} เป็น context หรือ surrounding words

skip-gram จะตรงข้ามกับ CBOW ก็คือจะใช้ target word เป็น input และทำนาย context words