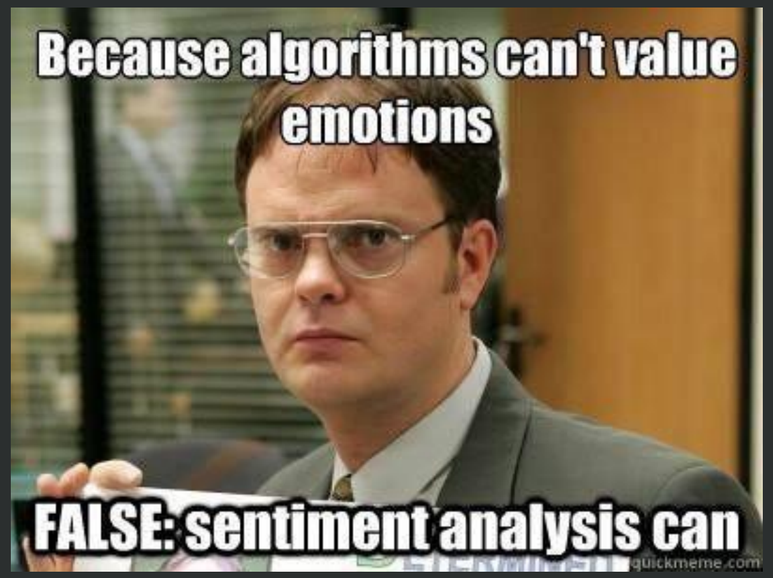
Introduction to NLP



How to process human language

so that your computer can understand it

But first...

What is Human Language?

It is a complex system of communication that allows people to express thoughts, emotions, and ideas through a structured combination of words and sentences.

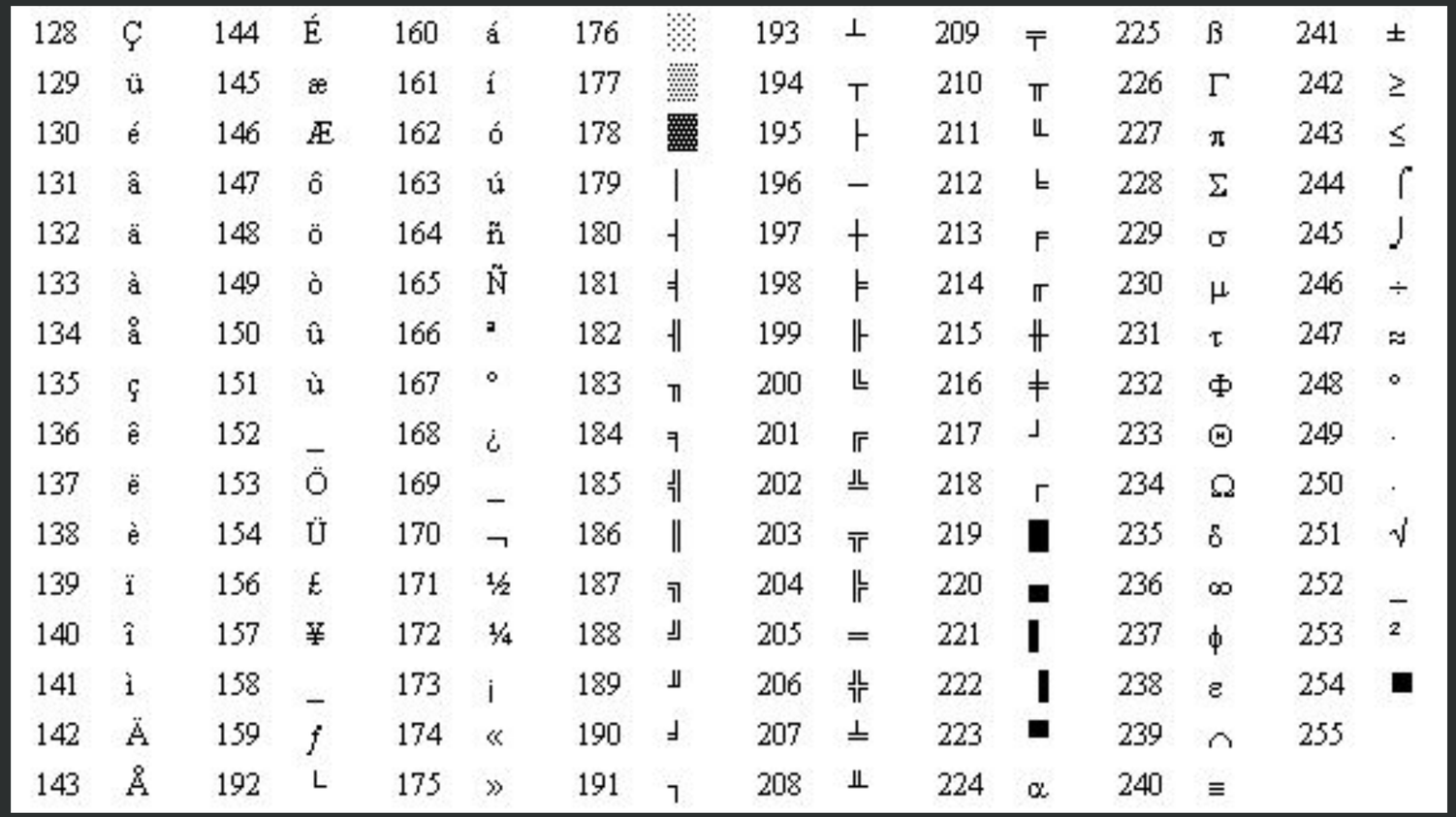
Human Language create an **infinite number of sentences** from a **finite set of elements**.

So...

How can our computer understand it?

We already know that we need to **code the language to numerical values** (as we do with categorical values, for example)

Let’s think about an approach to this problem...



128	Ç	144	É	160	á	176	▒	193	┴	209	╤	225	ß	241	±
129	ü	145	æ	161	í	177	▒	194	┬	210	╥	226	Γ	242	≥
130	é	146	Æ	162	ó	178	▓	195	├	211	╙	227	π	243	≤
131	â	147	ô	163	ú	179	│	196	─	212	╘	228	Σ	244	⌠
132	ä	148	ö	164	ñ	180	┤	197	┼	213	╒	229	σ	245	⌡
133	à	149	ò	165	Ñ	181	╡	198	╞	214	╓	230	µ	246	÷
134	å	150	û	166	ª	182	╢	199	╟	215	╫	231	τ	247	≈
135	ç	151	ù	167	º	183	╖	200	╚	216	╪	232	Φ	248	°
136	ê	152	_	168	¿	184	╕	201	╔	217	┘	233	Θ	249	·
137	ë	153	Ö	169	_	185	╣	202	╩	218	┌	234	Ω	250	·
138	è	154	Ü	170	¬	186	║	203	╦	219	▉	235	δ	251	√
139	ï	156	£	171	½	187	╗	204	╠	220	▄	236	∞	252	_
140	î	157	¥	172	¼	188	╝	205	═	221	▌	237	ϕ	253	²
141	ì	158	_	173	¡	189	╜	206	╬	222	▐	238	ε	254	■
142	Ä	159	ƒ	174	«	190	╛	207	╧	223	▀	239	∩	255	
143	Å	192	└	175	»	191	┐	208	╨	224	α	240	≡		

For example, we use ASCII to **storage characters** into our computers.

Understand this is important to understand the basics for our coding problem

Like ASCII, we could assign **numerical values** to words and **storage the results in a dictionary**

... But there is a problem

If we just gave numeric values to words, there’s no way our computer can interpreter these values the way we want it

=

15

=

20

=

30

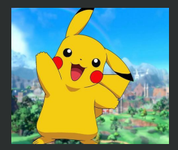
To our machine, Pikachu is the double of Plane and Car is halfway between the others

And for that...

We Vectorize (Tokenize) the words

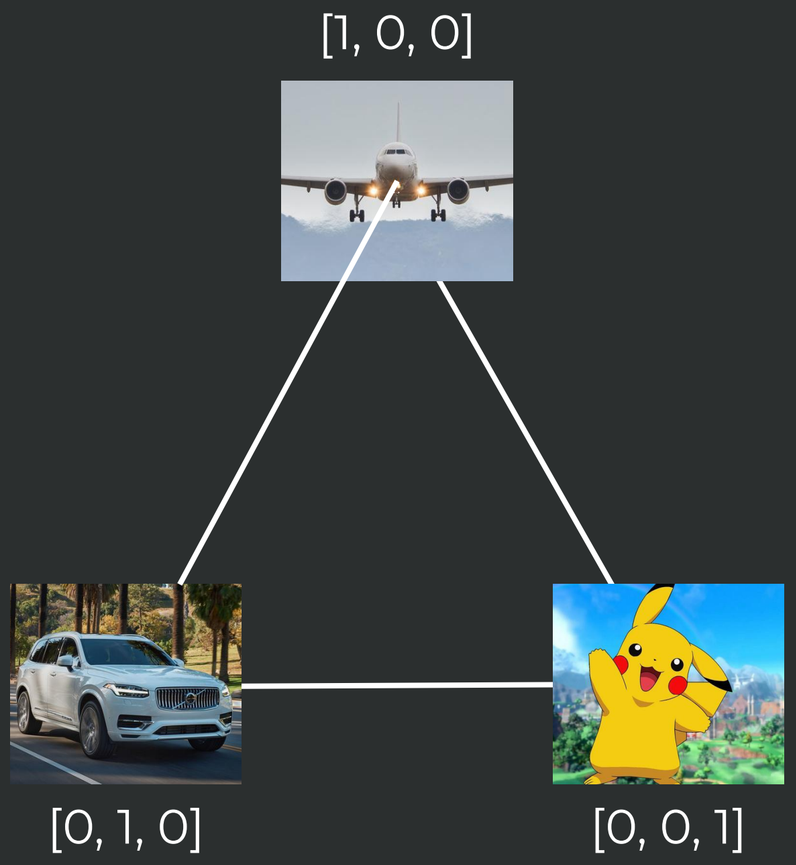
Airplane = 0 → [1, 0, 0]

Car = 1 → [0, 1, 0]

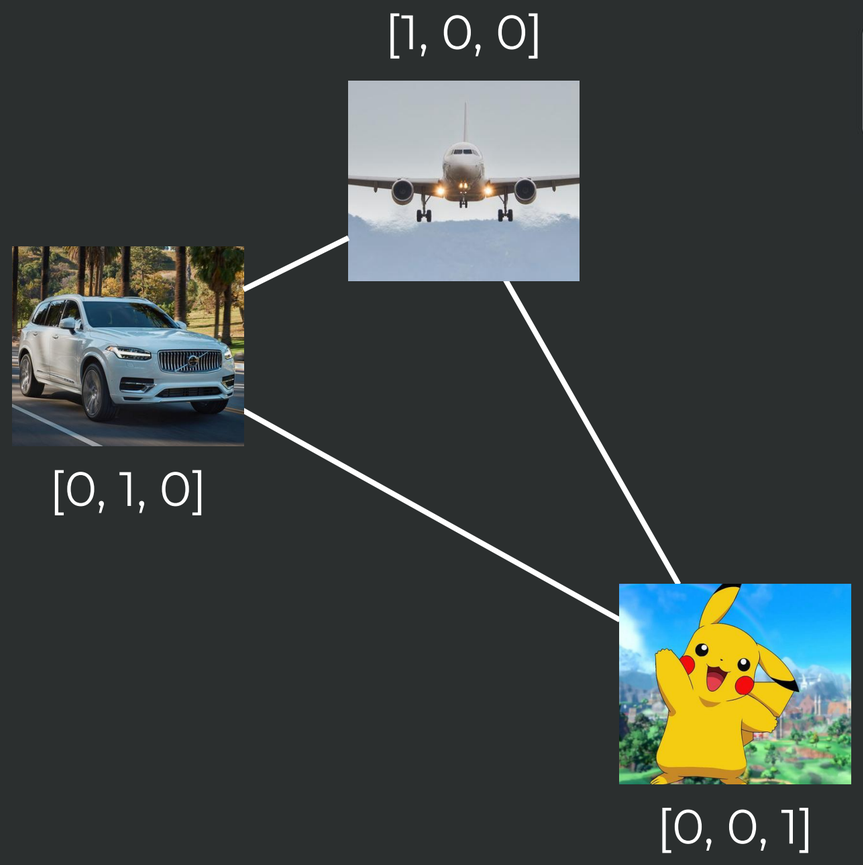


Pikachu = 2 → [0, 0, 1]

Now that we transform them into tokens, their values are **unique and at same distance**. But there is still a problem: to resolve x problems, we need to change the distances between them



All elements have their own dimension and same distance.

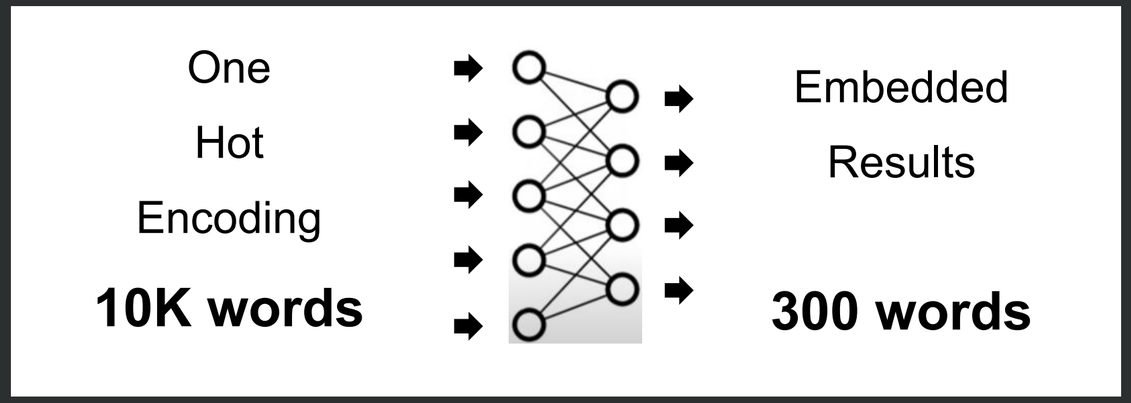


It makes more sense if the distance between some of them were shorter.

That's why...

We use Embeddings

These NLP techniques can **zip one-hot encoding** vectors to **reduce their dimensionality**



- Compact into smaller dimensions
- Can be pre-trained to group words by categories

NLP Embeddings can be combined with **Recurrent Neuronal Networks** for complex works like language translators or sentiment analysis...

But the old ones have a big problem...

Can’t recognise context...

And this is a problem, because human language is all about context. Think about sarcasm, old RNNs are sequential and can’t find the true intention on an ironic phrase.

wh

But one advanced architecture born to solve this problem and upgrade not only NLP, but also others fields like Computer Vision or Multimodal Tasks (text+image+audio)

TRANSFORMERS showed up as a deep learning revolution. This NN architecture has one crucial characteristic: **Self-Awareness**.

Self-Awareness can get **all inputs at once** instead of RNN sequential. This is used in NLP for be aware of the **position weight** in a sentence.