

Reprezentări numerice ale datelor în Machine Learning

Funcții matematice

Cum arată o funcție matematică?

a) Cu un singur parametru și un singur rezultat ($f: \mathbb{R} \rightarrow \mathbb{R}$)

$$f(x) = x^2 + 2x + 1$$

$$f(3) = 16$$

b) Cu mai mulți parametri și un singur rezultat ($f: \mathbb{R}^2 \rightarrow \mathbb{R}$)

$$f(x, y) = x^2 + y^2 + xy$$

$$f(2, 3) = 19$$

c) Cu un singur parametru și mai multe rezultate ($f: \mathbb{R} \rightarrow \mathbb{R}^2$)

$$f(x) = (x^3 + 3, x - 4)$$

$$f(2) = (11, -2)$$

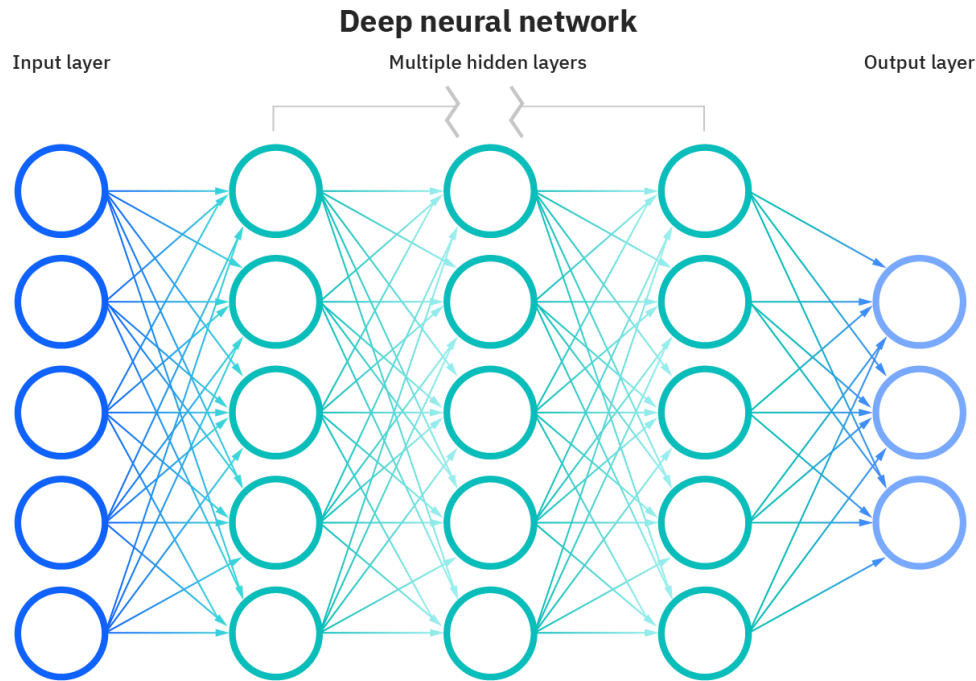
d) Cu mai mulți parametri și mai multe rezultate ($f: \mathbb{R}^n \rightarrow \mathbb{R}^k$)

$$f(x_1, x_2, \dots, x_n) = (f_1(x_1, x_2, \dots, x_n), \dots, f_k(x_1, x_2, \dots, x_n))$$

Legătura cu modelele de Machine Learning

Modelele de ML știu să lucreze doar cu numere!

Majoritatea modelelor de Machine Learning primesc datele într-o formă numerică, efectuează câteva operații matematice din care rezultă alte numere, care în cele din urmă sunt interpretate de programatori.

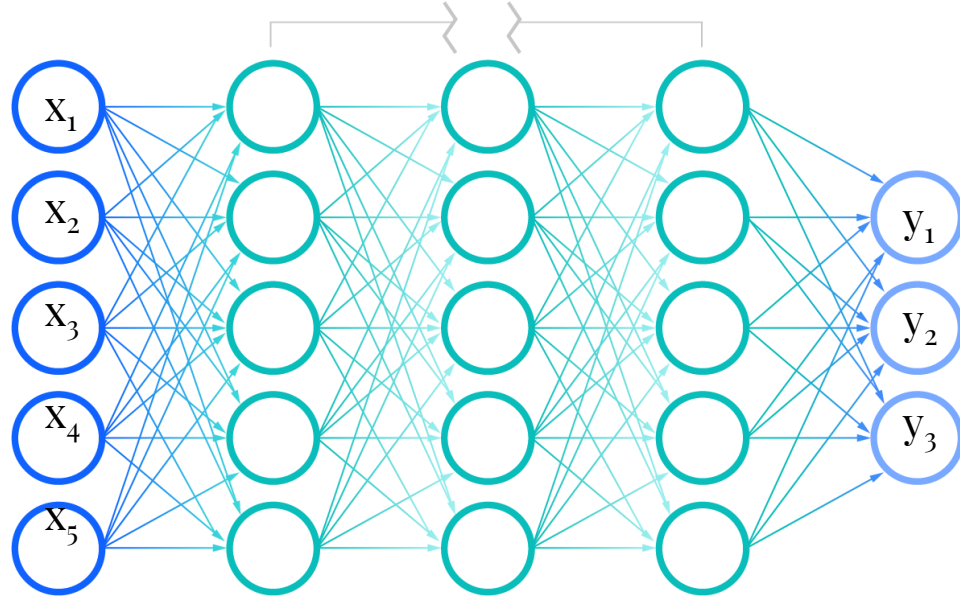


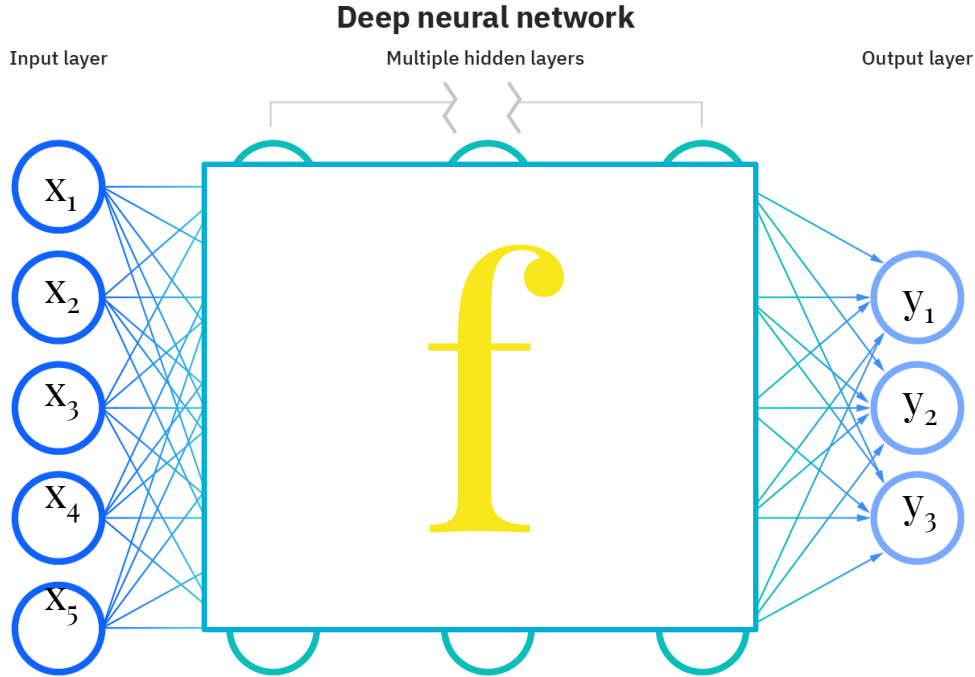
Deep neural network

Input layer

Multiple hidden layers

Output layer





$$f(x_1, x_2, x_3, x_4, x_5) = (y_1, y_2, y_3)$$

Cum putem reprezenta o rețea neuronală?

takes as input $x \in \mathbb{R}^n$ and gives an output $\in \mathbb{R}^m$

```
class MyNeuralNetwork {  
    float* w: [w1, w2, ..., wp]; # weights  
    float* b: [b1, b2, ..., bq]; # biases  
    function fArc:  $\mathbb{R}^{n+p+q} \rightarrow \mathbb{R}^m$  # the architecture  
    function loss:  $\mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$  # loss function  
  
    float* predict(x: [x1, x2, ..., xn]) {  
        return fArc(  
            x1, x2, ..., xn,  
            w1, w2, ..., wp,  
            b1, b2, ..., bq  
        );  
    }  
}
```

```
void train(x: [x1, x2, ..., xn], y: [y1, y2, ..., ym])  
{  
    float* pred = fArc(x, w, b);  
    float L = loss(pred, y);  
    # compute gradients,  $\delta w_i / \delta L$  and  $\delta b_j / \delta L$   
    # apply gradient descent on w and b  
}
```

† <https://youtu.be/VMj-3Sitkuo>

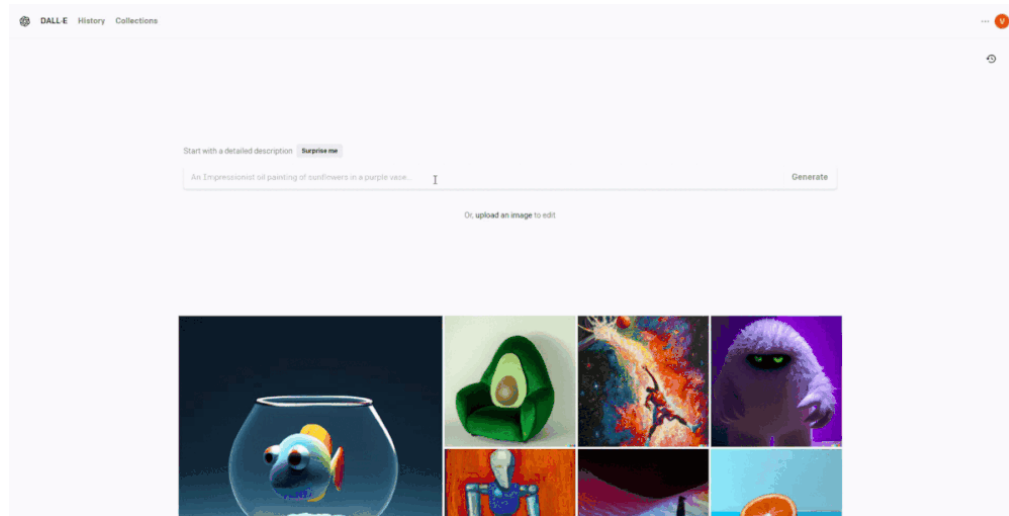
(Andrej Karpathy – building micrograd from scratch)

**Ce facem când datele cu care
lucrăm nu mai sunt numerice?**

DALL-E

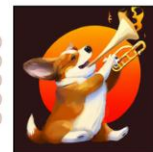
DALL-E este un model creat de cei de la OpenAI care traduce text în imagini corespunzătoare (există mai multe astfel de modele, acesta este unul dintre ele).

<https://labs.openai.com/>



"a corgi
playing a
flame
throwing
trumpet"

DALL-E




DALL-E

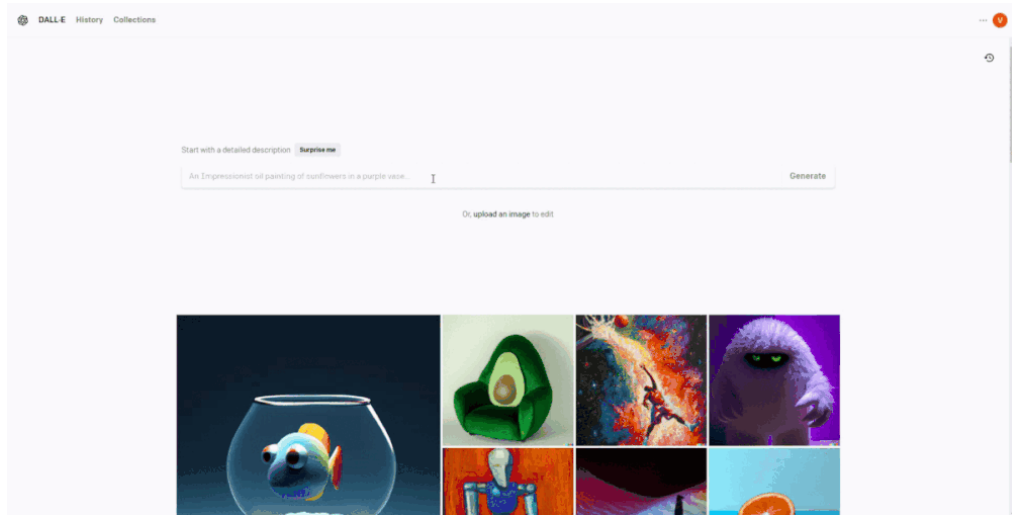
DALL-E este un model creat de cei de la OpenAI care traduce text în imagini corespunzătoare (există mai multe astfel de modele, acesta este unul dintre ele).

<https://labs.openai.com/>

Nu există funcție matematică care să primească text, nici care să scoată imagini, ce putem face?

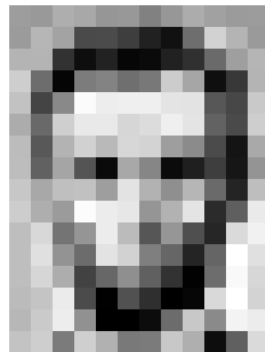
f("a corgi playing a flame throwing trumpet") ??

f() ??



Reprezentarea numerică a imaginilor

O imagine este doar o matrice formată din pixeli, deci, pentru imaginile alb-negru, putem vedea intensitatea fiecărui pixel ca output pentru funcția noastră. Imaginile colorate au pentru fiecare pixel 3 valori (RGB – Red, Green, Blue, câte o intensitate pentru fiecare), deci vom avea de 3 ori mai multe valori. De obicei aceste valori sunt numere naturale cuprinse între 0 și 255.



167	153	174	148	150	162	129	151	172	163	155	156
155	182	163	74	75	62	93	17	110	210	180	154
180	180	50	14	94	6	10	93	48	106	159	181
206	109	5	124	131	111	120	204	166	15	55	180
194	68	137	251	237	239	228	227	87	71	201	
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	216	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	35	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	96	103	143	95	90	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

167	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	93	17	110	210	180	154
180	180	50	14	94	6	10	93	48	106	159	181
206	109	5	124	131	111	120	204	166	15	55	180
194	68	137	251	237	239	228	227	87	71	201	
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	35	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	96	103	143	95	90	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

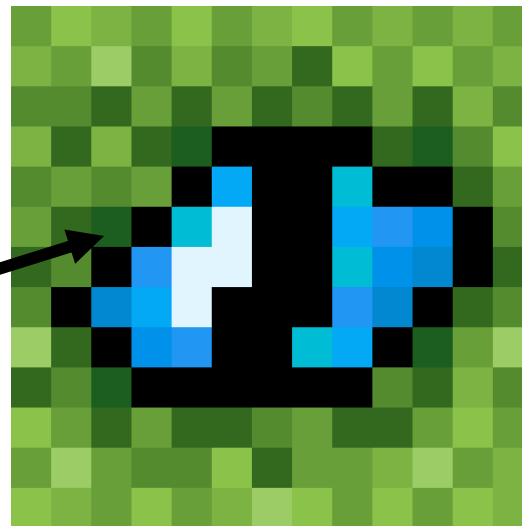


Reprezentarea numerică a imaginilor

Cod HEX: #1B5E20

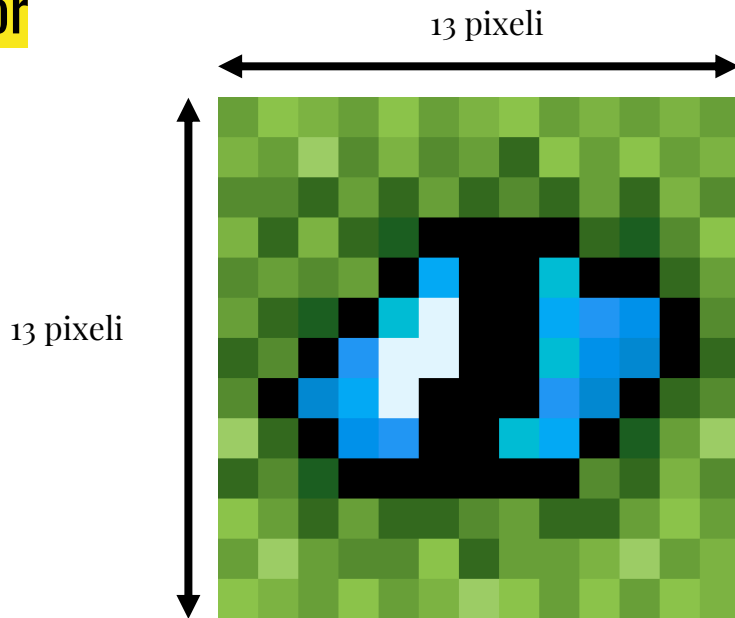
Intensități:

- Roșu: 27
- Verde: 94
- Albastru: 32



Reprezentarea numerică a imaginilor

În total vom avea $13 \times 13 = 169$ de pixeli, iar cum fiecare pixel conține 3 valori (intensitatea pentru roșu, cea pentru verde și cea pentru albastru), vom avea în total $3 * 169 = 507$ valori (numere întregi cuprinse între 0 și 255).



Reprezentarea numerică a imaginilor

DALL-E generează imagini cu dimensiunea de 1024×1024 pixeli, deci va fi nevoie să genereze în final 2^{20} pixeli, adică $3 * 2^{20} = 3,145,728$ valori.



“Teddy bears working on new AI research underwater with 1990s technology”



“An armchair in the shape of an avocado”

Reprezentarea numerică a imaginilor

DALL-E generează imagini cu dimensiunea de 1024×1024 pixeli, deci va fi nevoie să genereze în final 2^{20} pixeli, adică $3 \times 2^{20} = 3,145,728$ valori.

Deci, până acum, funcția matematică pentru modelul nostru de generat imagini din text ar arăta în stilul acesta:

$$f(?) = (y_1, y_2, \dots, y_{3,145,728})$$

Mai este nevoie să găsim o reprezentare a textului în numere.



“Teddy bears working on new AI research underwater with 1990s technology”

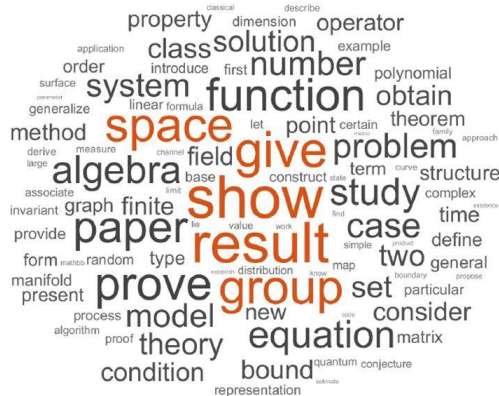


“An armchair in the shape of an avocado”

Reprezentarea numerică a textului

Acest task este puțin mai dificil, fiind nevoie să păstrăm cumva o legătură între cuvinte. De exemplu, nouă ne este ușor să găsim o asociere între cuvintele “părinte” și “tată”, însă pentru calculator este un task mai dificil.

O variantă ar fi să ținem toate cuvintele într-o listă, astfel vom avea un index pentru fiecare și aceea ar fi reprezentarea numerică, însă așa nu ar fi nicio legătură între cuvinte, și modelul de Machine Learning ar trebui să învețe el acest lucru → va trebui antrenat mai mult, deci nu este cea mai fiabilă variantă.



For the n^{th} word in the vocabulary, the n^{th} element is 1.

<i>artificial</i>	=	[1 0 0 0 0 0 0 0 0 0]
<i>be</i>	=	[0 1 0 0 0 0 0 0 0 0]
<i>branch</i>	=	[0 0 1 0 0 0 0 0 0 0]
<i>computer</i>	=	[0 0 0 1 0 0 0 0 0 0]
<i>enable</i>	=	[0 0 0 0 1 0 0 0 0 0]
<i>human</i>	=	[0 0 0 0 0 1 0 0 0 0]
<i>intelligence</i>	=	[0 0 0 0 0 0 1 0 0 0]
<i>language</i>	=	[0 0 0 0 0 0 0 1 0 0]
<i>natural</i>	=	[0 0 0 0 0 0 0 0 1 0]
<i>process</i>	=	[0 0 0 0 0 0 0 0 0 1]

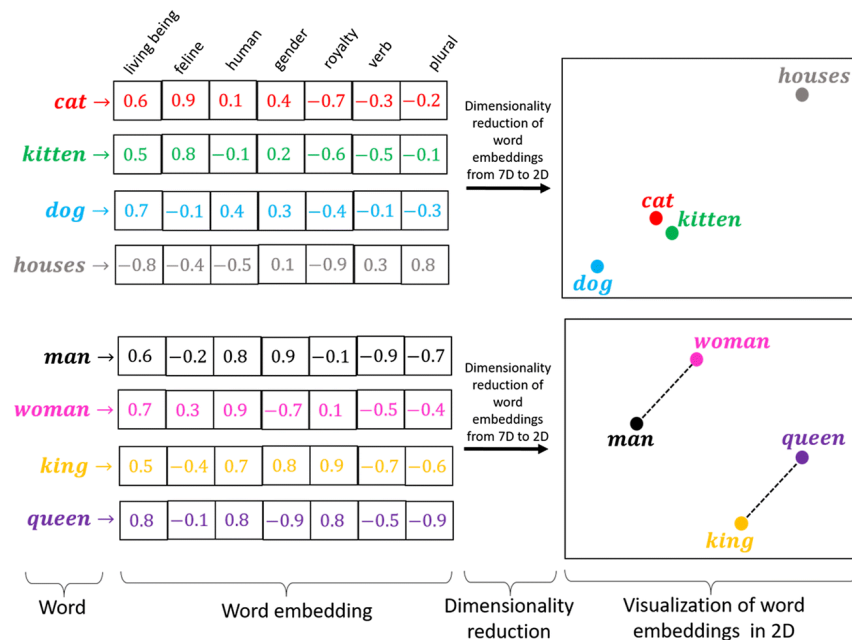
Reprezentarea numerică a textului

De aceea, au apărut câteva modele de Machine Learning care se ocupă cu partea de creat embedding-uri (perechi de numere reprezentative) pentru cuvinte.

Modelele respective au învățat să găsească relații între cuvinte, acestea putând fi interpretate și de noi.

Putem considera *Emb* un astfel de model, rezultatul aplicării acestui algoritm asupra unui cuvânt o să fie o pereche de numere:

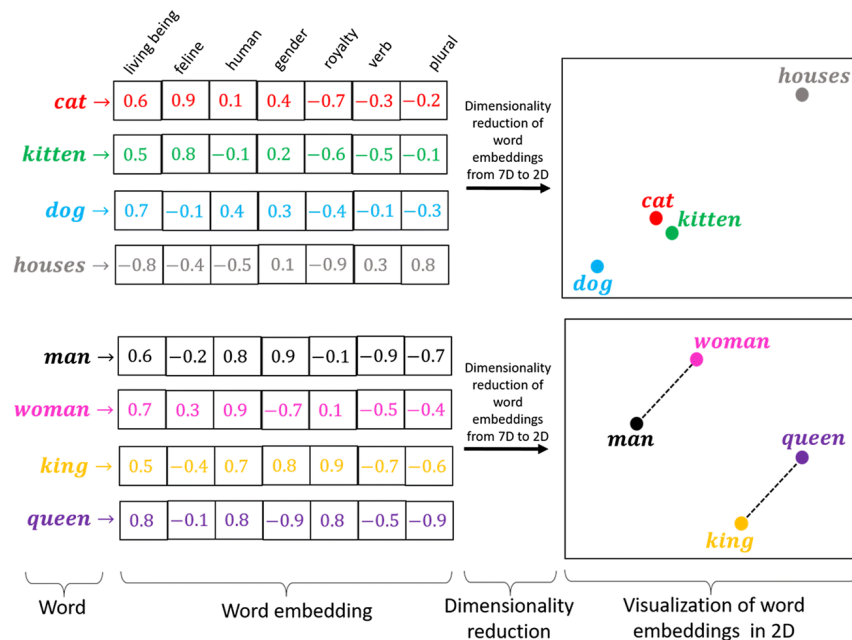
$$Emb(\text{"word"}) = (x_1, x_2, \dots, x_k)$$



Reprezentarea numerică a textului

Din figura din dreapta, partea de sus, putem observa că perechile de numere rezultate pentru cuvintele “cat” și “kitten” sunt mai apropiate între ele decât de alte cuvinte, precum “dog” și “houses”, relație pe care și noi, ca oameni o putem observa.

=> cuvintele asemănătoare au reprezentări asemănătoare



Reprezentarea numerică a textului

În partea de jos a figurii, putem observa următoarele lucruri:

$$Emb(\text{"man"}) = (0.6, -0.2, 0.8, 0.9, -0.1, -0.9, -0.7)$$

$$Emb(\text{"woman"}) = (0.7, 0.3, 0.9, -0.7, 0.1, -0.5, -0.4)$$

$$Emb(\text{"man"}) - Emb(\text{"woman"})$$

$$= (-0.1, -0.5, -0.1, 1.6, -0.2, -0.4, -0.3)$$

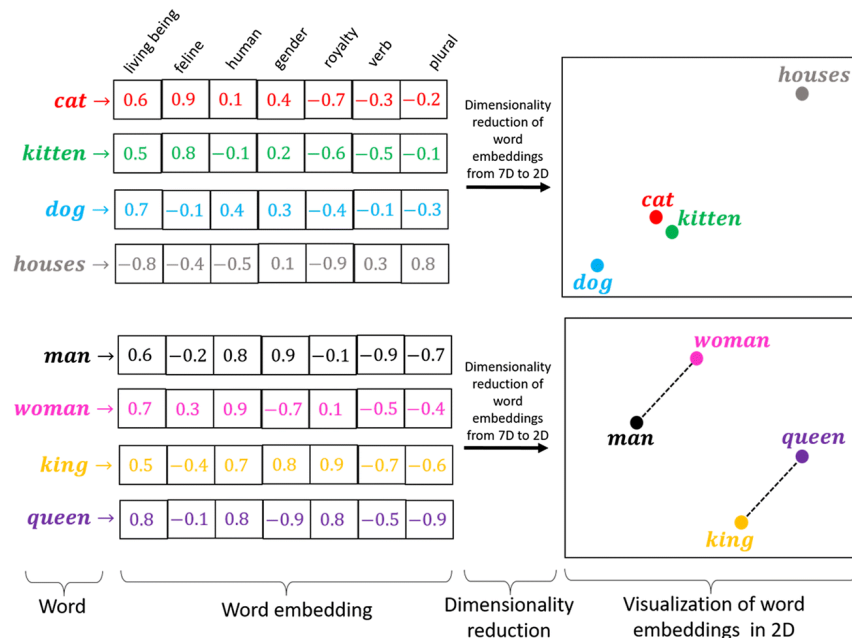
$$Emb(\text{"king"}) = (0.5, -0.4, 0.7, 0.8, 0.9, -0.7, -0.6)$$

$$Emb(\text{"queen"}) = (0.8, -0.1, 0.8, -0.9, 0.8, -0.5, -0.9)$$

$$Emb(\text{"king"}) - Emb(\text{"queen"})$$

$$= (-0.3, -0.3, -0.1, 1.7, 0.1, -0.2, 0.3)$$

Comparând cele 2 diferențe, putem observa că sunt destul de apropiate, însemnând că algoritmul a învățat că diferența dintre “man” și “woman” este asemănătoare cu cea dintre “king” și “queen”.



Menti: 3804 9730
<https://menti.com>

word2vec

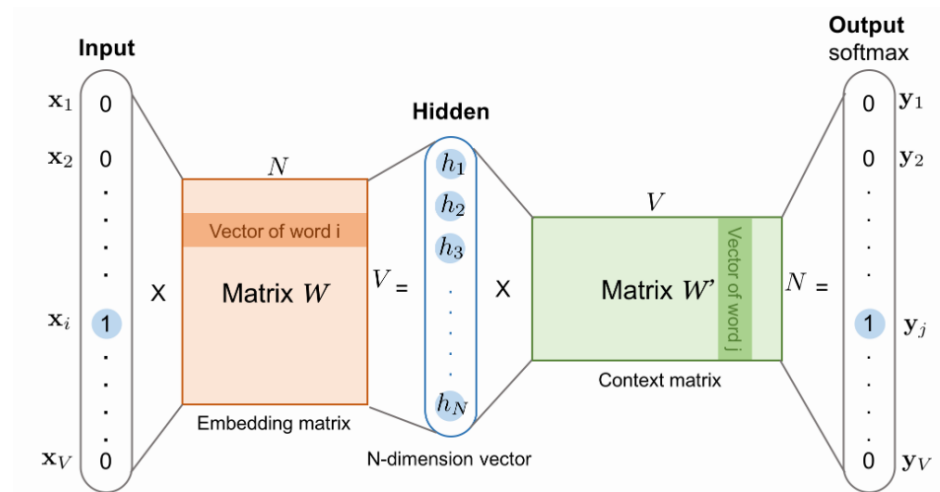


Table 1: Examples of five types of semantic and nine types of syntactic questions in the Semantic-Syntactic Word Relationship test set.

Type of relationship	Word Pair 1		Word Pair 2	
Common capital city	Athens	Greece	Oslo	Norway
All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
Currency	Angola	kwanza	Iran	rial
City-in-state	Chicago	Illinois	Stockton	California
Man-Woman	brother	sister	grandson	granddaughter
Adjective to adverb	apparent	apparently	rapid	rapidly
Opposite	possibly	impossibly	ethical	unethical
Comparative	great	greater	tough	tougher
Superlative	easy	easiest	lucky	luckiest
Present Participle	think	thinking	read	reading
Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
Past tense	walking	walked	swimming	swam
Plural nouns	mouse	mice	dollar	dollars
Plural verbs	work	works	speak	speaks

Table 8: Examples of the word pair relationships, using the best word vectors from Table 4 (Skip-gram model trained on 783M words with 300 dimensionality).

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

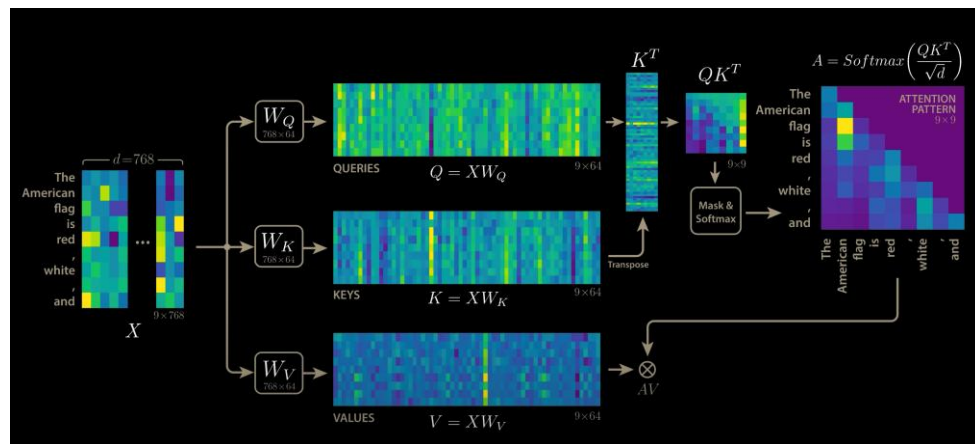
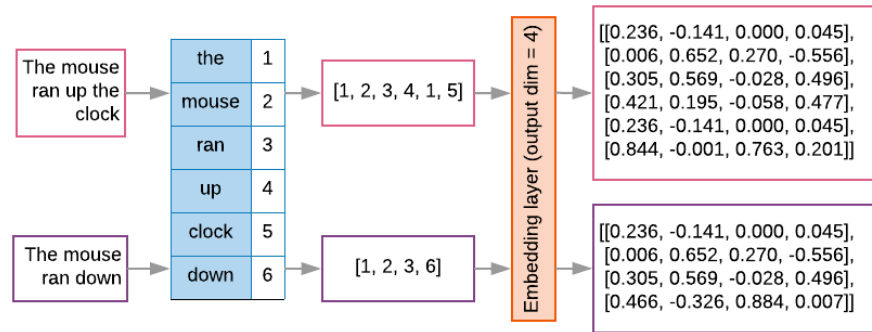
Cum lucrăm cu propoziții?

Problemă: ne este imposibil să ținem câte un embedding pentru fiecare propoziție.

Soluție: vom împărți propozițiile în mai multe părți (tokens), mulțimea tuturor părților va forma un vocabular și vom reține embedding-uri doar pentru fiecare token din acel vocabular.

Problemă: fiecare propoziție va avea un număr variabil de tokens, cum putem să le transmitem funcției noastre?

Soluție: modele care lucrează cu date secvențiale (RNN, LSTM, Transformer, etc.)



How DeepSeek rewrote the Transformer - https://youtu.be/oVLAoVGf_74

Transformers explained visually (3B1B) - <https://youtu.be/wjZofJXov4M>

Este suficient?

În momentul de față, funcția noastră matematică pentru modelul de generat imagini din text arată astfel:

input: text

$(t_1, t_2, \dots, t_q) = \text{Tokenize}(\text{text})$

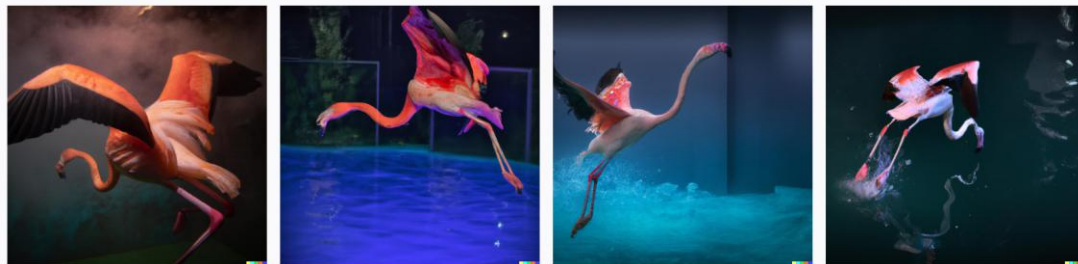
$(x_{1,1..k}, x_{2,1..k}, \dots, x_{q,1..k}) = \text{map}(\text{Emb}, [t_1, t_2, \dots, t_q])$

$(y_1, y_2, \dots, y_n) = f(x_{1,1..k}, x_{2,1..k}, \dots, x_{q,1..k})$

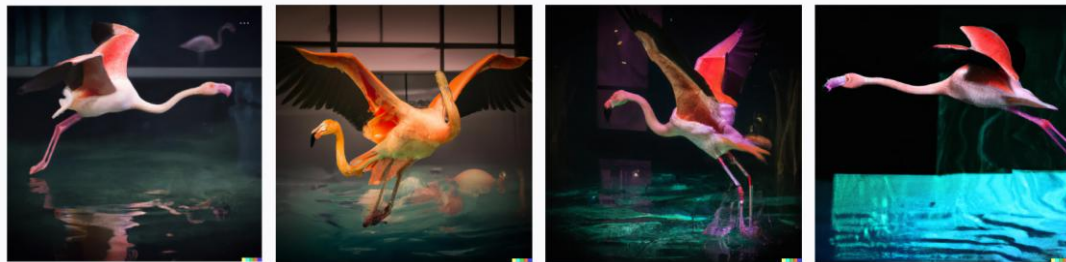
image = *imageFromVector*(y_1, y_2, \dots, y_n)

output: image

Să nu uităm că *Emb* și *f* sunt funcții matematice!



“flying flamingo in an aquarium”

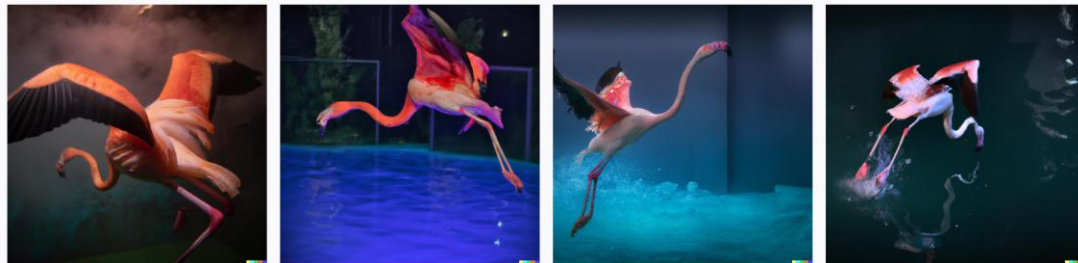


“flying flamingo in an aquarium”

Funcțiile matematice au mereu același rezultat!

$$f(x) = x^2 + 2x + 1$$

$f(3) = 16$ în orice moment, nu se va schimba niciodată (funcțiile matematice sunt pure), dar totuși DALL-E produce output-uri diferite pentru același text. De ce?



“flying flamingo in an aquarium”



“flying flamingo in an aquarium”

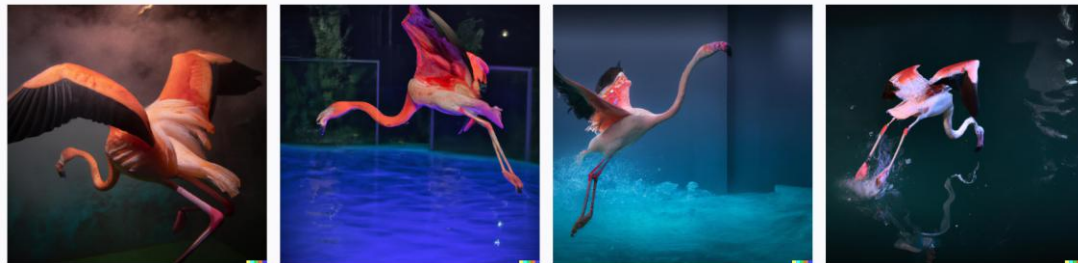
Spațiul latent

Pe lângă text-ul dat ca input, de multe ori, algoritmi generativi primesc și un set de numere aproximativ aleator (z_1, z_2, \dots, z_p), astfel că, la fiecare generare, fiecare imagine va arăta diferit.

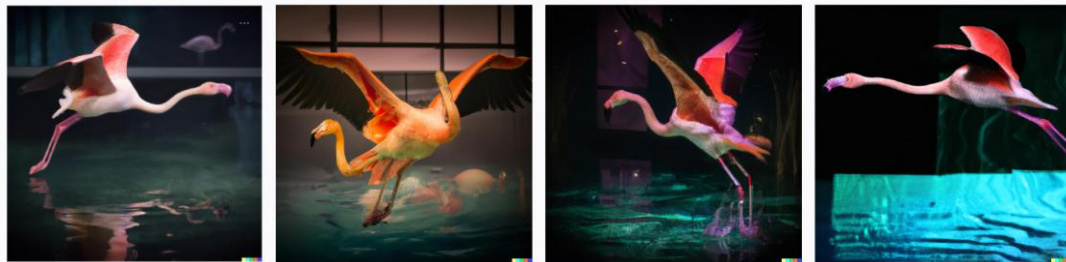
Intuiție: $f(x) = x^2 + 2x + 1$

$f(3) = 16$; $f(3.01) = 16.08$; $f(3.02) = 16.16$

➔ modificări mici input conduc la modificări mici ale rezultatului pentru funcțiile continue



“flying flamingo in an aquarium”

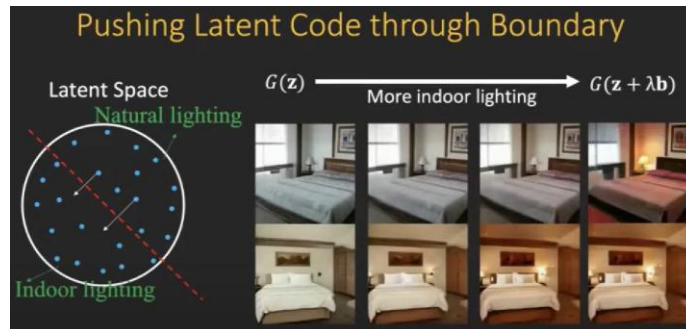


“flying flamingo in an aquarium”

Spațiul latent

Pe lângă text-ul dat ca input, de multe ori, algoritmi generativi primesc și un set de numere aproximativ aleator (z_1, z_2, \dots, z_p), astfel că, la fiecare generare, fiecare imagine va arăta diferit.

Un lucru interesant observat, este că aceste valori au anumite caracteristici asupra imaginii finale, deci, modificând câte puțin unul dintre numerele date aleator pentru o imagine (ex. z_i), o caracteristică din imaginea rezultat se va modifica tot câte puțin (asemănător cu rețelele de embedding de la traducerea textului).



Adding clouds



Adding vegetation



Sursa: <https://youtu.be/8Hm4ad5QIUE>

Reprezentarea matematică finală

Deci în final, modelul la care am reușit să ajungem este o funcție matematică:

input: text

$(z_1, z_2, \dots, z_p) = \text{generateRandomNumbers}()$

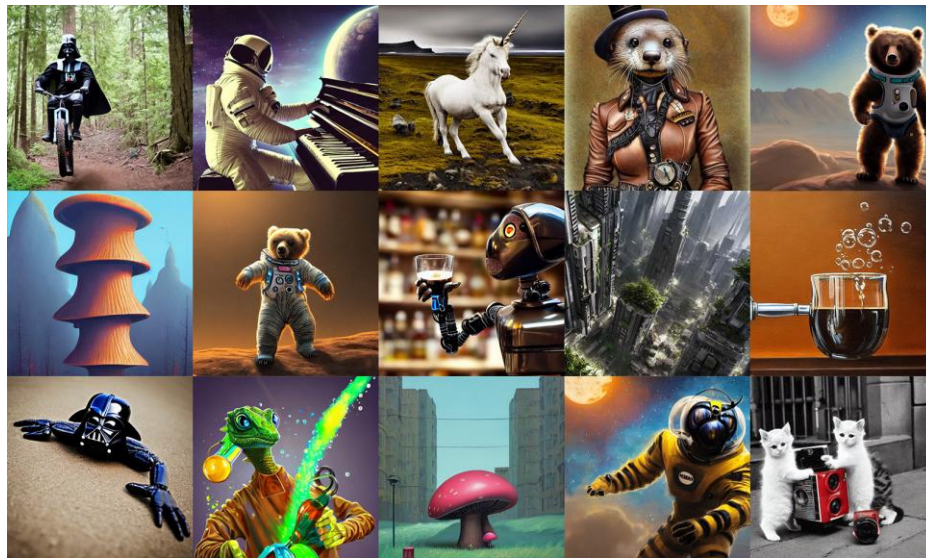
$(t_1, t_2, \dots, t_q) = \text{Tokenize}(\text{text})$

$(x_{1,1,k}, x_{2,1,k}, \dots, x_{q,1,k}) = \text{map}(\text{Emb}, [t_1, t_2, \dots, t_q])$

$(y_1, y_2, \dots, y_k) = f(x_{1,1,k}, x_{2,1,k}, \dots, x_{q,1,k}, z_1, z_2, \dots, z_p)$

$\text{image} = \text{imageFromVector}(y_1, y_2, \dots, y_k)$

output: image



Reprezentarea matematică finală

Deci în final, modelul la care am reușit să ajungem este o funcție matematică:

input: text

$(z_1, z_2, \dots, z_p) = \text{generateRandomNumbers}()$

$(t_1, t_2, \dots, t_q) = \text{Tokenize}(\text{text})$

$(x_{1,1,k}, x_{2,1,k}, \dots, x_{q,1,k}) = \text{map}(\text{Emb}, [t_1, t_2, \dots, t_q])$

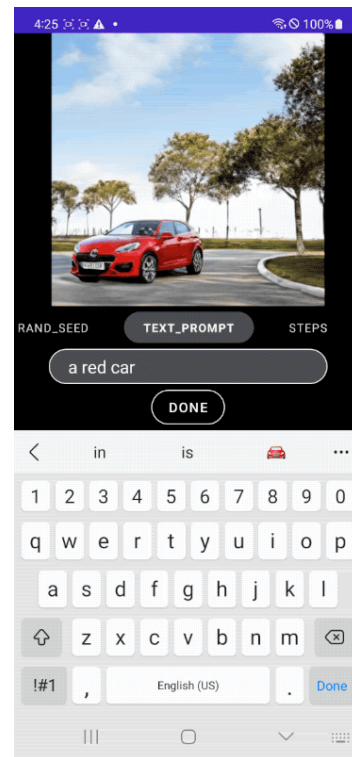
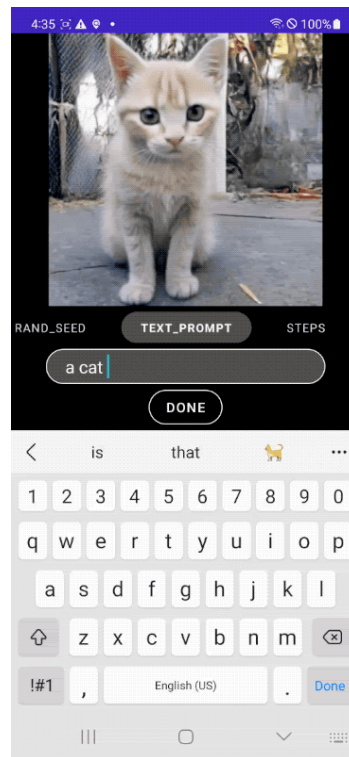
$(y_1, y_2, \dots, y_k) = f(x_{1,1,k}, x_{2,1,k}, \dots, x_{q,1,k}, z_1, z_2, \dots, z_p)$

$\text{image} = \text{imageFromVector}(y_1, y_2, \dots, y_k)$

output: image

Partea ușoară: $\text{generateRandomNumbers}()$,
 $\text{Tokenize}()$, $\text{Emb}()$ (există deja soluții),
 $\text{imageFromVector}()$

Partea dificilă: găsirea funcției f



<https://research.google/blog/mobilediffusion-rapid-text-to-image-generation-on-device/>

Reprezentarea matematică finală

Deci în final, modelul la care am reușit să ajungem este o funcție matematică:

input: text

$(z_1, z_2, \dots, z_p) = \text{generateRandomNumbers}()$

$(t_1, t_2, \dots, t_q) = \text{Tokenize}(\text{text})$

$(x_{1,1,k}, x_{2,1,k}, \dots, x_{q,1,k}) = \text{map}(\text{Emb}, [t_1, t_2, \dots, t_q])$

$(y_1, y_2, \dots, y_k) = f(x_{1,1,k}, x_{2,1,k}, \dots, x_{q,1,k}, z_1, z_2, \dots, z_p)$

$\text{image} = \text{imageFromVector}(y_1, y_2, \dots, y_k)$

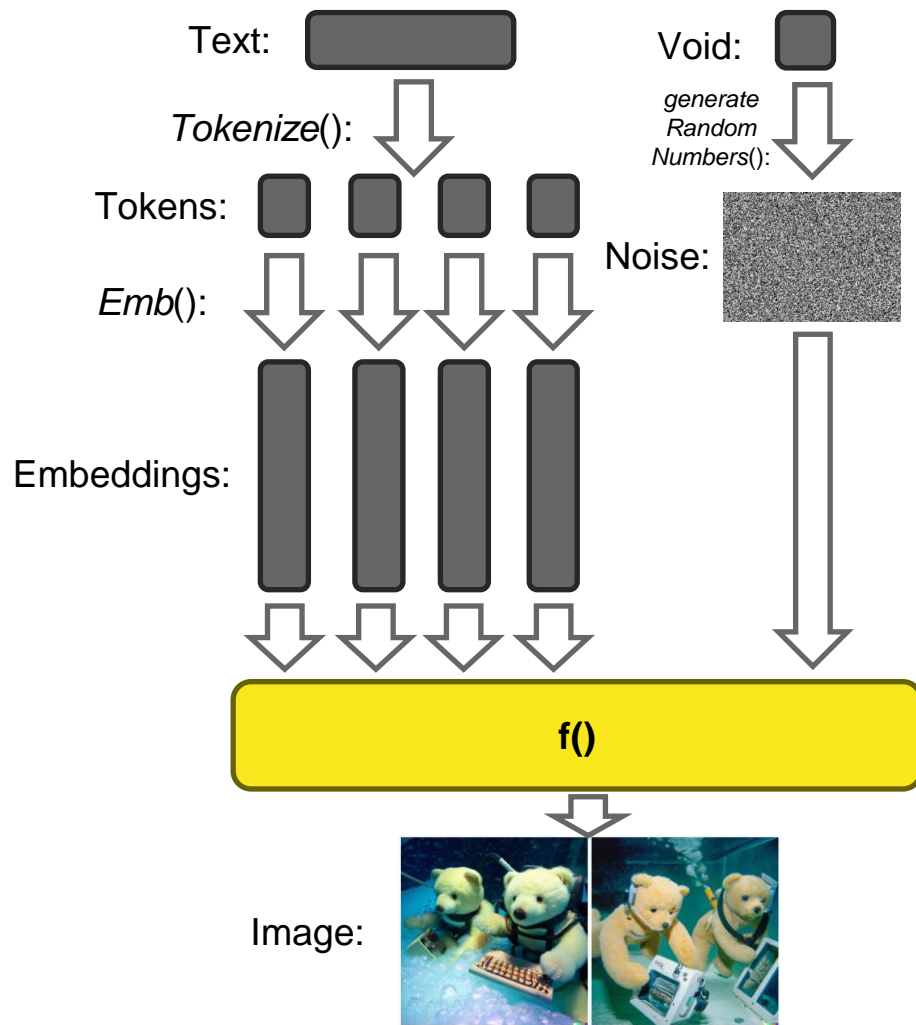
output: image

Partea ușoară: $\text{generateRandomNumbers}()$,

$\text{Tokenize}()$, $\text{Emb}()$ (există deja soluții),

$\text{imageFromVector}()$

Partea dificilă: găsirea funcției f



That's it.
Mersi de participare!

Concepte cheie:

- Algoritmii de ML lucrează cu numere → Trebuie să transformăm datele de prelucrat în numere.
- Majoritatea algoritmilor de ML sunt funcții matematice, învățarea constă în găsirea acestora.
- Cele mai semnificative reprezentări numerice ale unor date păstrează relații între date pe care putem într-o oarecare măsură să le interpretăm și noi.