Recuperación de Información Multimedia

Deep Learning (texto, multimodal)

CC5213 – Recuperación de Información Multimedia
Departamento de Ciencias de la Computación
Universidad de Chile
Juan Manuel Barrios – https://juan.cl/mir/ – 2020

Vectorización

- Para poder utilizar texto en una red neuronal es necesario “vectorizar” el texto
- Enfoque tradicional:
  - Bag-of-Words
  - Stemming, Lematización, Stop Lists
- Redes Neuronales:
  - One-hot encoding
  - Word embedding

One-Hot Encoding

- Primero se define el vocabulario (lista de palabras conocidas)
- Para un vocabulario de *n* palabras, la *i*-ésima palabra se codifica con un vector de *n* dimensiones, con un 1 en la *i*-ésima coordenada y 0 en el resto:

  (0,…,0,1,0,…0)
- Es una codificación sparse (muchos ceros)
- Alta dimensionalidad
- Todas las palabras son igualmente distintas

Word Embeddings

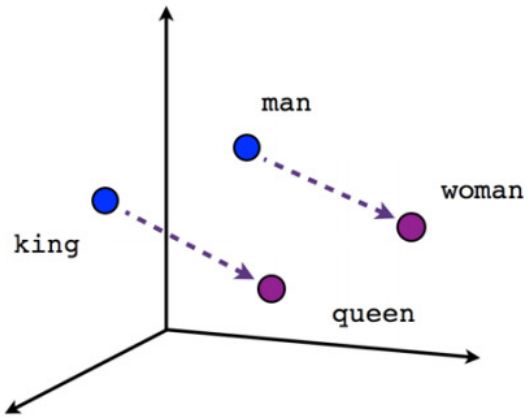
- Representar palabras con vectores cuya distancia se ajuste a su diferencia en significado
- Es una codificación densa
- Menor dimensionalidad que one-hot (ej.: 300-d)
- Similitud entre palabras se debe a similitudes en su contexto
- Se entrena una conversión desde vectores one-hot a vectores densos usado una MLP
  - Es posible usar vectores pre-entrenados para vocabularios conocidos o entrenarlos para cada problema a resolver

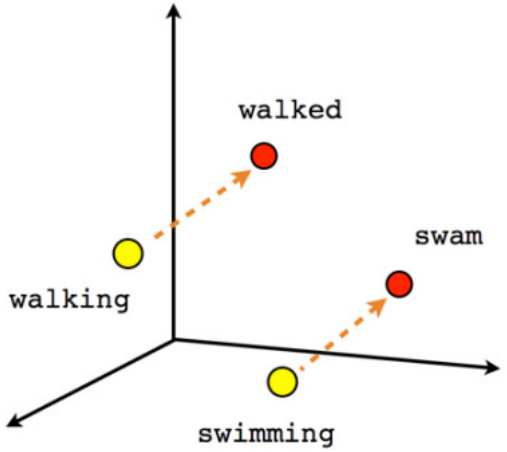
Word Embedding Space

- Se espera que el espacio de las palabras tenga propiedades como:
  - Palabras que son sinónimos estén asociadas a vectores muy cercanos entre si (distancia euclidiana cercana a cero)
  - Las direcciones en el espacio tengan algún significado de operación con las palabras:
    - singular a plural, masculino a femenino, sustantivo a adverbio, infinitivo a participio, presente a pasado, etc.
    - “el día soleado” ↔ “los días soleados”
    - “el gato negro” ↔ “la gata negra”
- El espacio de las palabras depende del idioma y también del uso (legal, técnico, popular)

Word Embedding Space

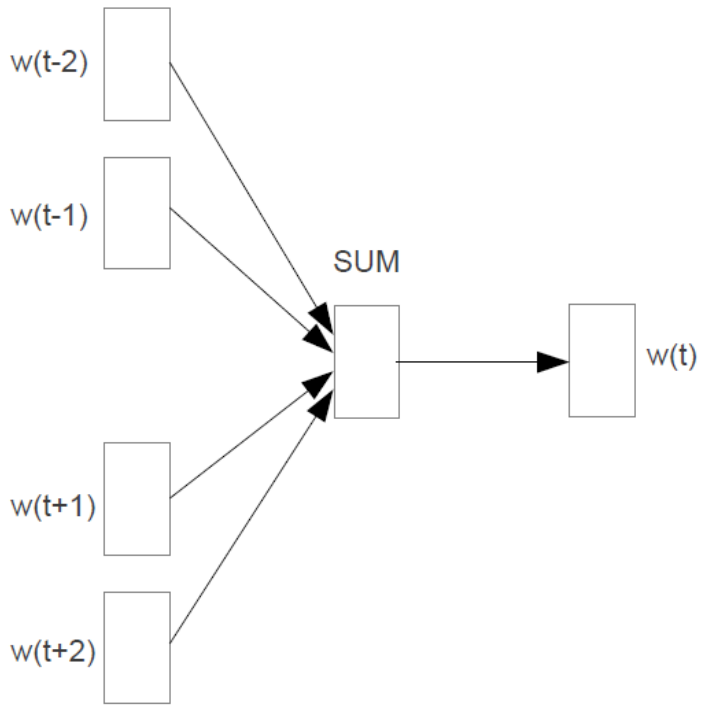
- Permiten resolver analogías:
  - (sintáctico) “Aparente” es a “Aparentemente” como “Evidente” es a …
  - (semántico) “Atenas” es a “Grecia” como “Oslo” es a …





Entrenamiento Word2Vec

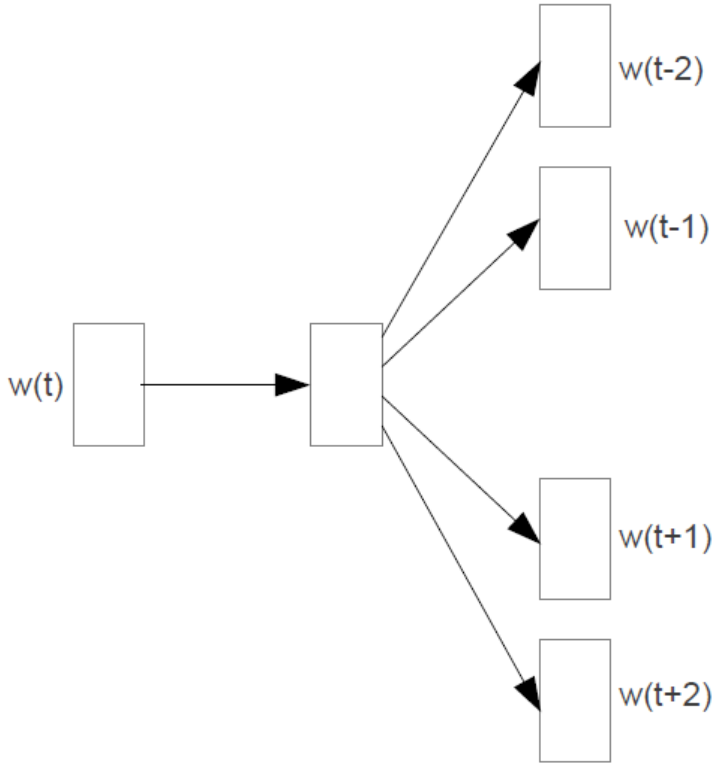
- Modelo Continuous Bag-of-Words
  - Predecir una palabra dadas sus palabras de contexto
  - Produce vectores con mejor resultado en predicción sintáctica



Mikolov et al. “Efficient Estimation of Word Representations inVector Space”. 2013. https://arxiv.org/pdf/1301.3781.pdf

Entrenamiento Word2Vec

- Modelo Continuous Skip-gram
  - Dada una palabra predecir sus palabras de contexto
  - Produce vectores con mejor resultado en predicción semántica



Mikolov et al. “Efficient Estimation of Word Representations inVector Space”. 2013. https://arxiv.org/pdf/1301.3781.pdf

GloVe

- https://nlp.stanford.edu/projects/glove/
- Se basa en factorizar una matriz de co-ocurrencia de palabras
- Muy similar en idea a Latent Semantic Analysis
  - Ver capítulo de Bag-of-Words y LSA

FastText

- https://github.com/facebookresearch/fastText
- Calcula vectores para secuencias de caracteres y los suma para crear el vector de cada palabra
- Permite generar un vector para palabras desconocidas

Sentence Embedding

- Calcular un vector para una frase.
  - Promedio de los word vectors
    - Eliminar stop-words
  - Promedio ponderado por IDF

$$v_s = \frac{1}{|s|} \sum_{w \in s} \text{IDF}_w v_w \qquad \text{IDF}_w := \log \frac{1+N}{1+N_w}$$

- Doc2Vec: Entrenar word2vec incluyendo un id del documento (sentence)

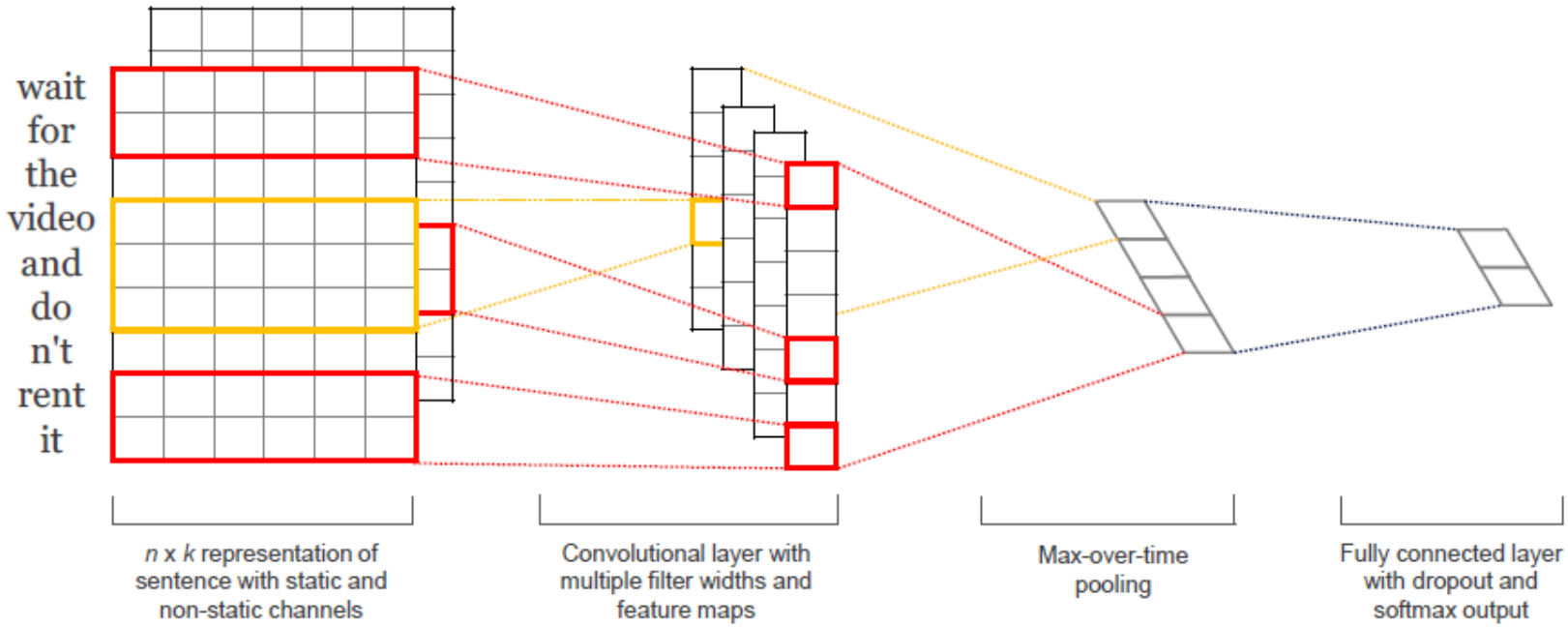
Arora et al. “A simple but tough-to-beat baseline for Sentence Embeddings”. 2017.
Le, Mikolov. “Distributed Representations of Sentences and Documents”. 2018.

Otros Vector Embeddings

- **Node2Vec**: Vectorizar un grafo calculando un vector por nodo al medir nodos vecinos
- **Item2Vec**: Calcular vectores de ítems para sistemas recomendadores

CNN para Texto

- Conv1D es similar a N-Grams



Kim. “Convolutional Neural Networks for Sentence Classification”. 2014

CNN para Texto

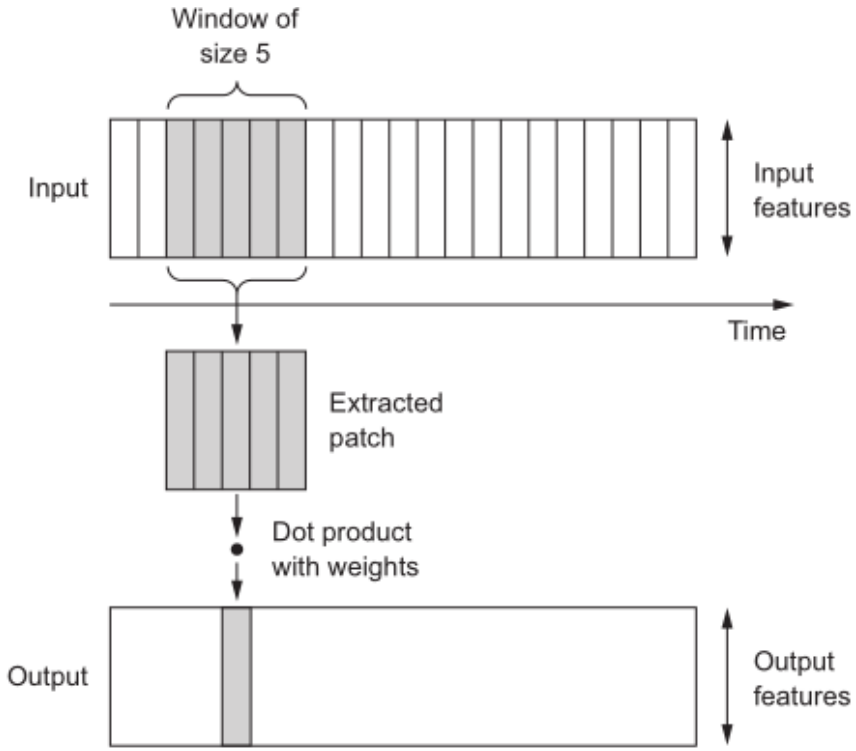
- Input: *n* palabras, cada palabra es un vector del word embedding de dim *k* (ej. k=300)
- Convolución: Un filtro de tamaño h corresponde un vector de h*k que se usa como producto punto con una ventana de h palabras

$$c_i = f(\mathbf{w} \cdot \mathbf{x}_{i:i+h-1} + b) \qquad \mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}]$$

- Max-Pooling en el tiempo $\hat{c} = \max\{\mathbf{c}\}$
- Se concatenan varios filtros para formar un vector
- 100 filtros de tamaño 3, 4, 5

CNN para Texto

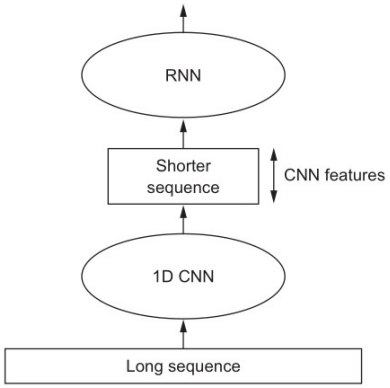
- Convolución 1D es el producto punto entre vectores



Ver Chollet, pag 225

CNN para Texto

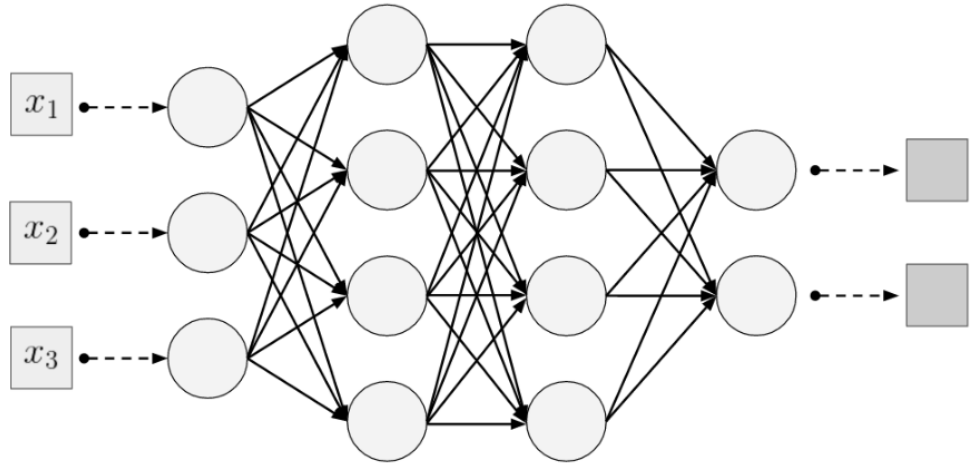
- Conv1D permiten hacer detección de grupos de palabras como n-grams
- Las convoluciones no ven la secuencias en el tiempo
- El mejor resultado se obtiene con una primera capa de Conv1D y luego una red recurrente que vea la secuencia temporal.

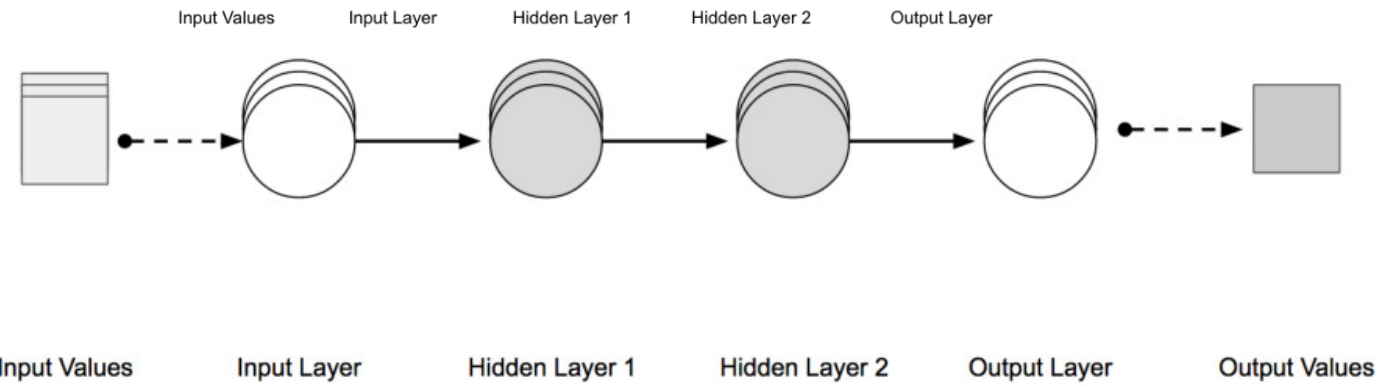


Ver Chollet, pag 229

Redes Recurrentes

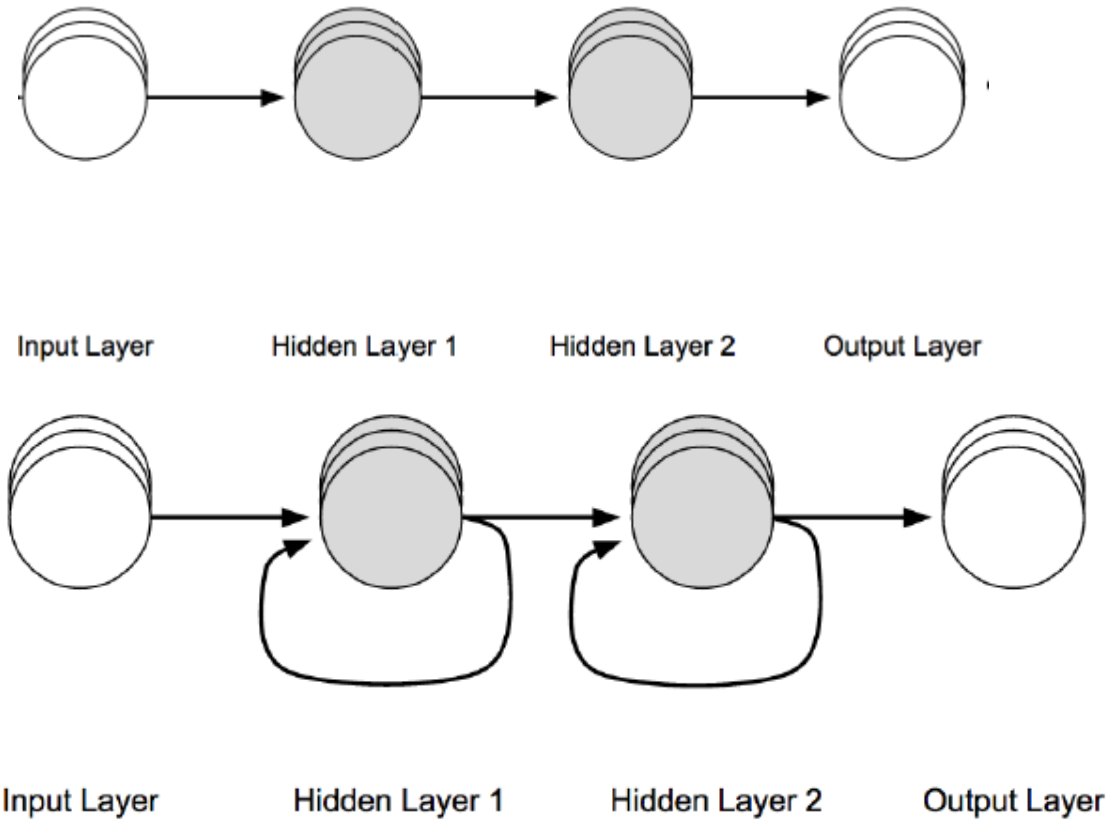
- Feed-Forward vs Recurrent





Redes Recurrentes

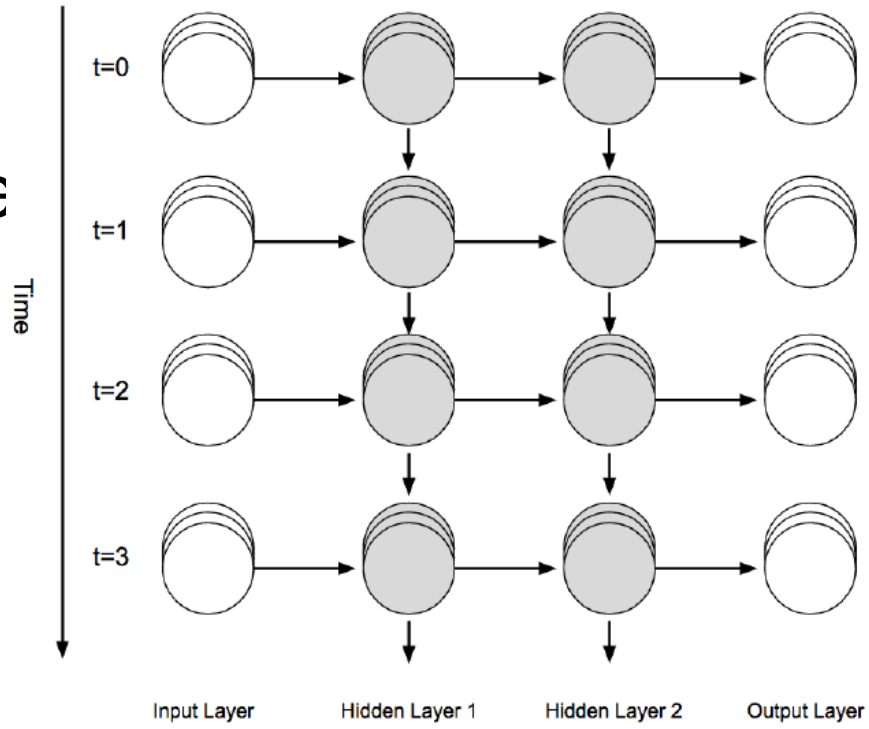
- Feed-Forward vs Recurrent



Ver Patterson, pag 152

RNN

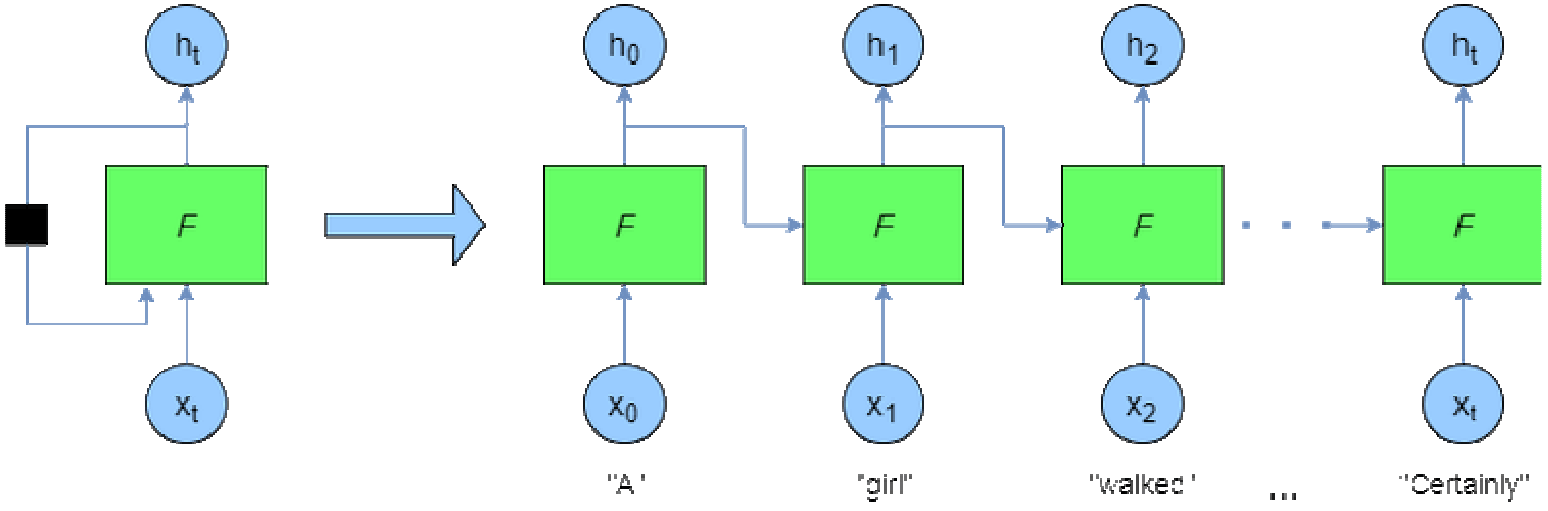
- Redes Recurrentes se usan para procesar datos con dimensión temporal, donde importa el orden de los datos
- RNN simples (“vanilla”) sufren del Vanishing Gradient para mantener información entre varias ventanas



Ver Patterson, pag 152

RNN

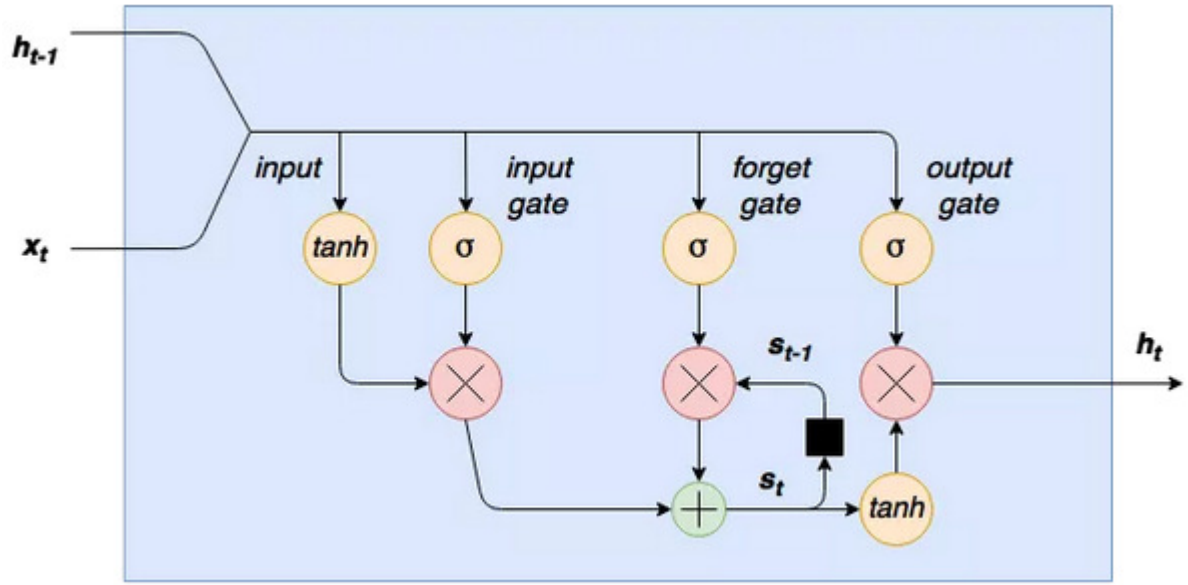
- El input de la red es una secuencia de vectores x_0 a x_t de la misma dimensión que se consumen uno a uno
- Cada entrada produce una salida intermedia h_i
- La entrada es el vector x_i junto con el estado anterior h_{i-1}
- El output final de la red es el estado final h_t



https://adventuresinmachinelearning.com/keras-lstm-tutorial/

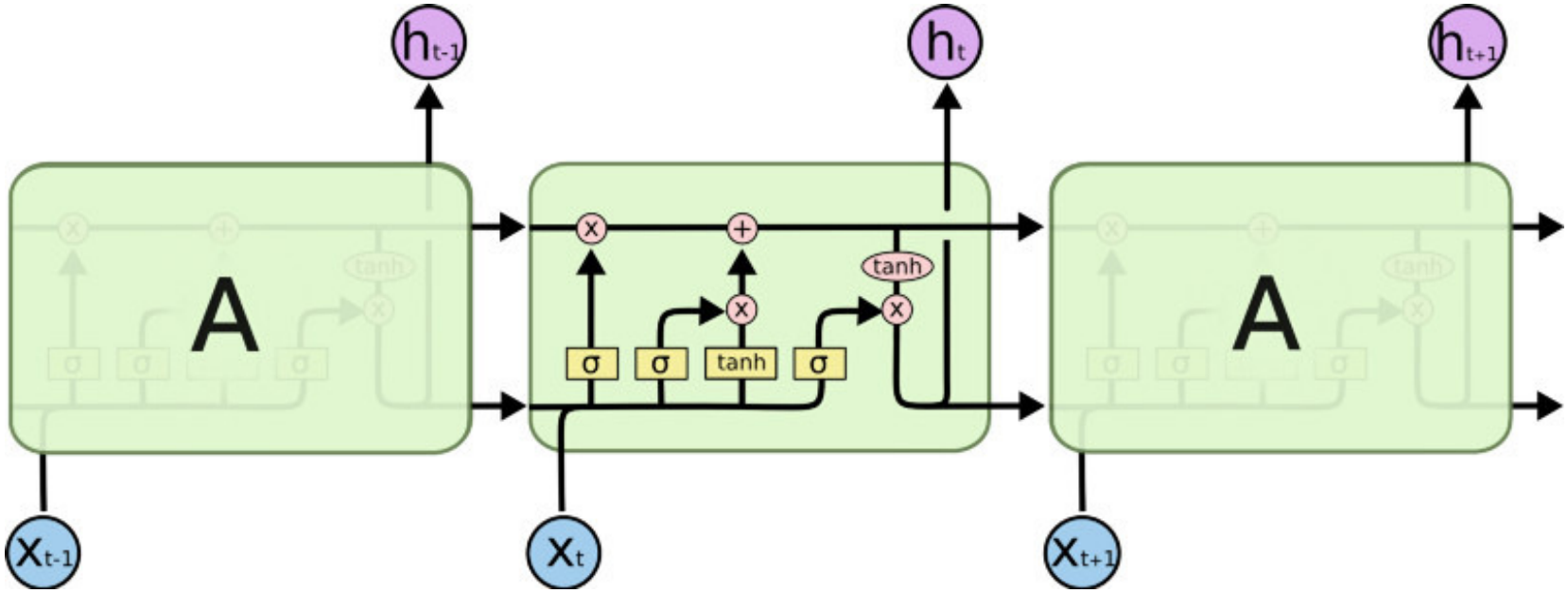
Long-Short Term Memory (LSTM)

- Contiene gates para decidir qué valores se leen del estado anterior y se generan a la salida (tanh)



LSTM

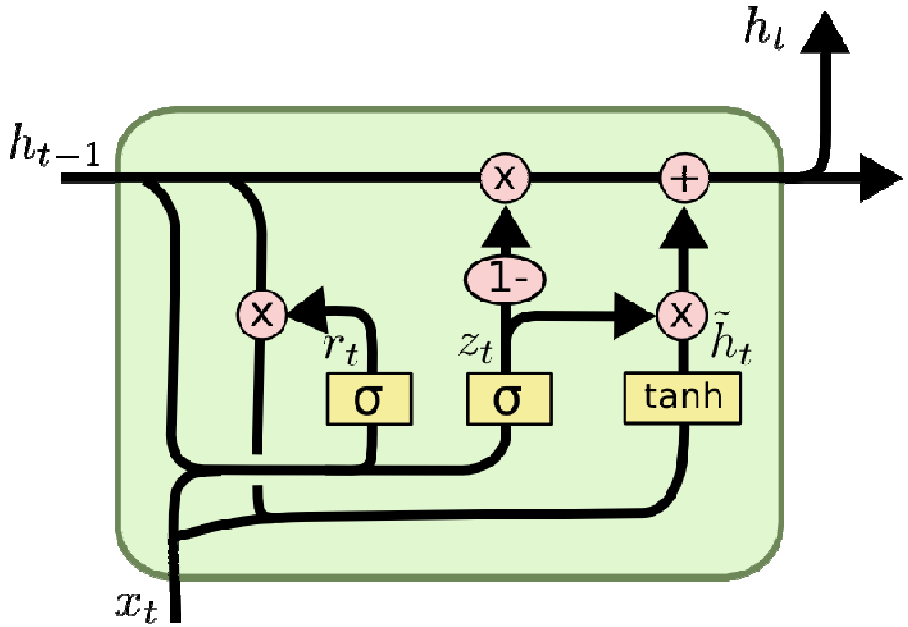
- Long-Short Term Memory puede guardar información por periodos largos y cortos gracias a las compuertas para guardar/olvidar



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

RNN

- GRU (Gated Recurrent Unit)
  - Variante de LSTM con menos gates
  - Es más simple y rápida de entrenar



Generación de texto con RNN

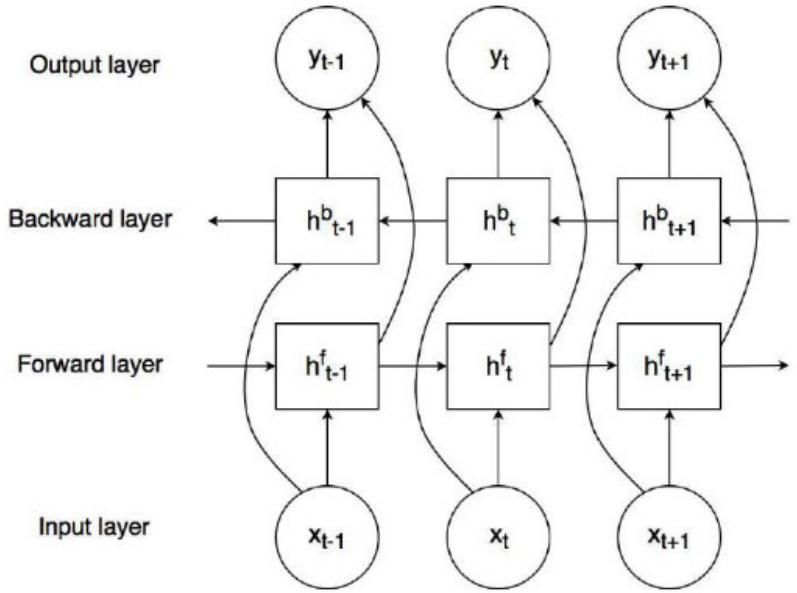
- Se debe entrenar con secuencias de largo fijo con cada valor de entrada y su valor de salida correspondiente
  - Por ejemplo, para entrenar una red que genere texto se usa:

Entradas		Salidas
[puedo,escribir,los,versos]		[más]
[escribir,los,versos,más]	→	[tristes]
[los,versos,más,tristes]		[esta]
[versos,más,tristes,esta]		[noche]

Ver Chollet, pag 271

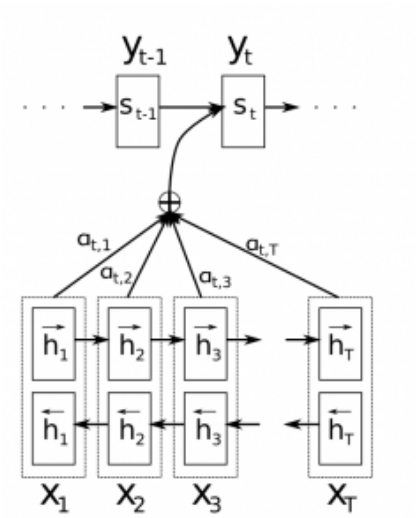
Bi-Direccional

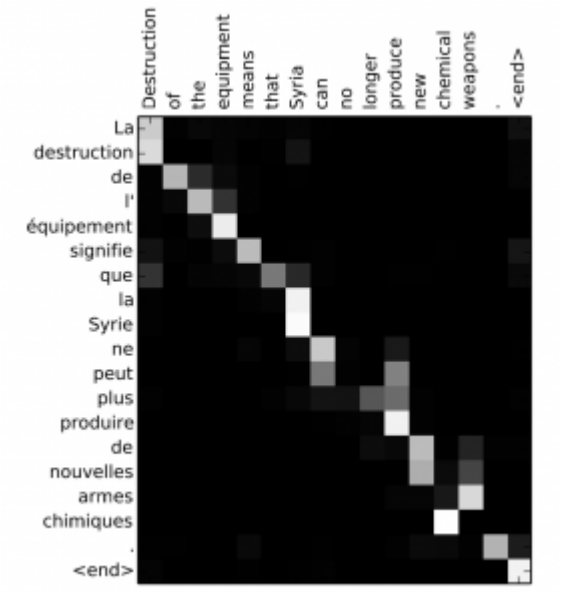
- Para reducir la influencia de los últimos valores en el resultado final
- Se concatenan las salidas de ambas direcciones



RNN con Zona de Atención

- Crear una zona donde se guarda la relación entre inputs y outputs
- Permite encontrar la causa de una decisión





by *ent270* , *ent223* updated 9:35 am et , mon march 2 , 2015 (*ent223*) *ent63* went familial for fall at its fashion show in *ent231* on sunday , dedicating its collection to `` mamma '' with nary a pair of `` mom jeans '' in sight . *ent164* and *ent21* , who are behind the *ent196* brand , sent models down the runway in decidedly feminine dresses and skirts adorned with roses , lace and even embroidered doodles by the designers ' own nieces and nephews . many of the looks featured saccharine needlework phrases like `` i love you ,

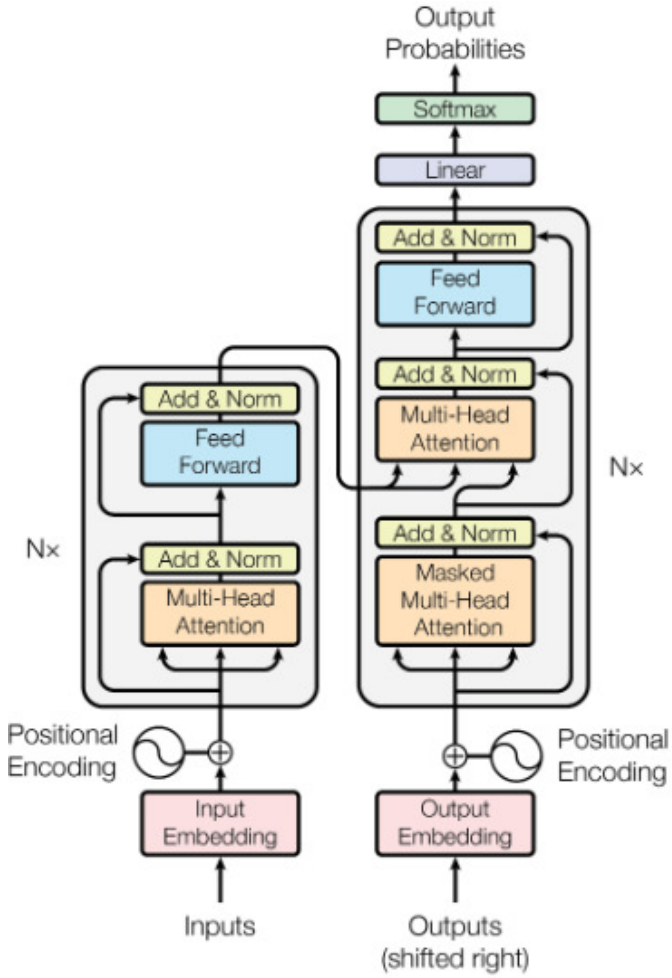
. . .

X dedicated their fall fashion show to moms

http://www.wildml.com/2016/01/attention-and-memory-in-deep-learning-and-nlp/

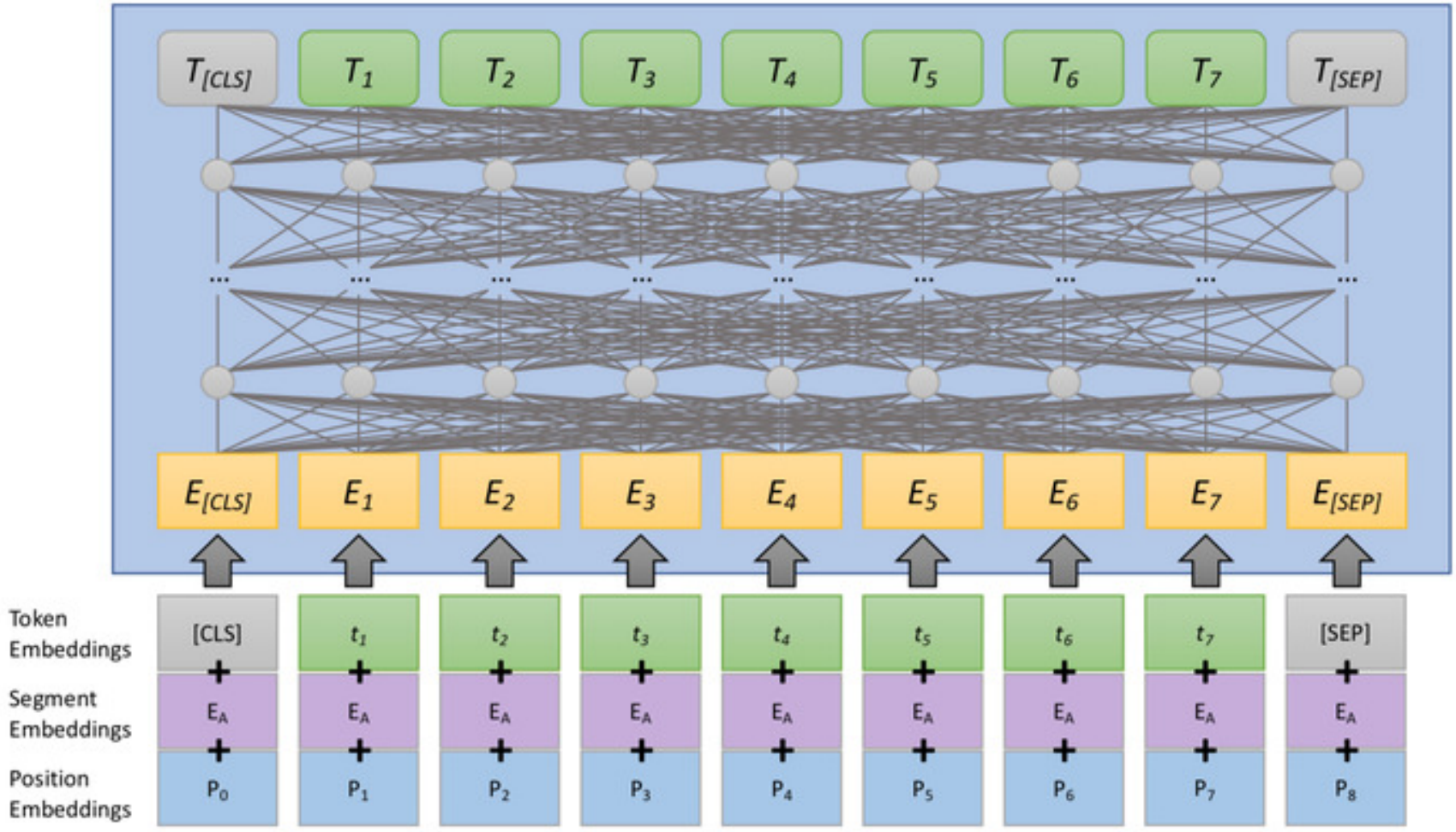
Self Attention

- Nueva arquitectura que reemplaza las RNNs
- Guardar la relación entre los datos de input con los mismos datos de input
- Permite encontrar relación entre palabras de la frase de entrada (ej. sujetos implícitos)



Vaswani et al. “Attention Is All You Need”. 2017.
https://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

BERT



Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. 2019. https://arxiv.org/pdf/1810.04805.pdf

Combinación Texto con Imágenes

Combinar Texto con Imágenes

- Dataset COCO tiene ~80 mil imágenes cada una con 5 descripciones (~400 mil descripciones)

•A white lawn chair laying on top of a sandy beach

•A sun shade sitting out on the beach

•Empty beach chair under an umbrella while different boats are out in the ocean

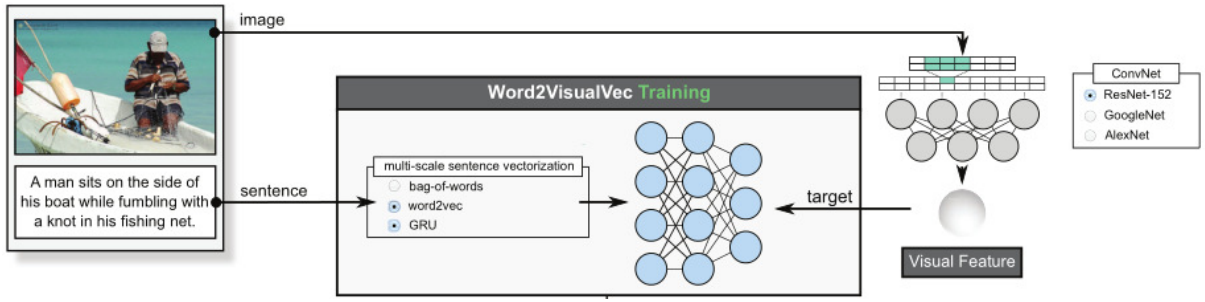
•A group of people are surfing and swimming in the ocean

•Sunsets over a surfer and other people enjoying the ocenn beach

•A child walking and watching a surfer at sunset

Buscar imágenes sin etiquetar

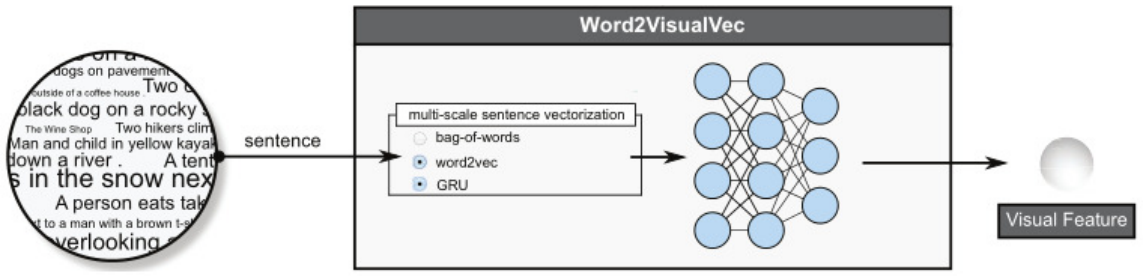
- Con datos de entrenamiento (COCO) calcular un vector visual para cada imagen y un vector textual para su descripción correspondiente
- Entrenar MLP para regresión, con entradas los vectores textuales y salidas los vectores visuales correspondientes
  - Conversión de espacios de características textual a visual (embedding)



Word2VisualVec

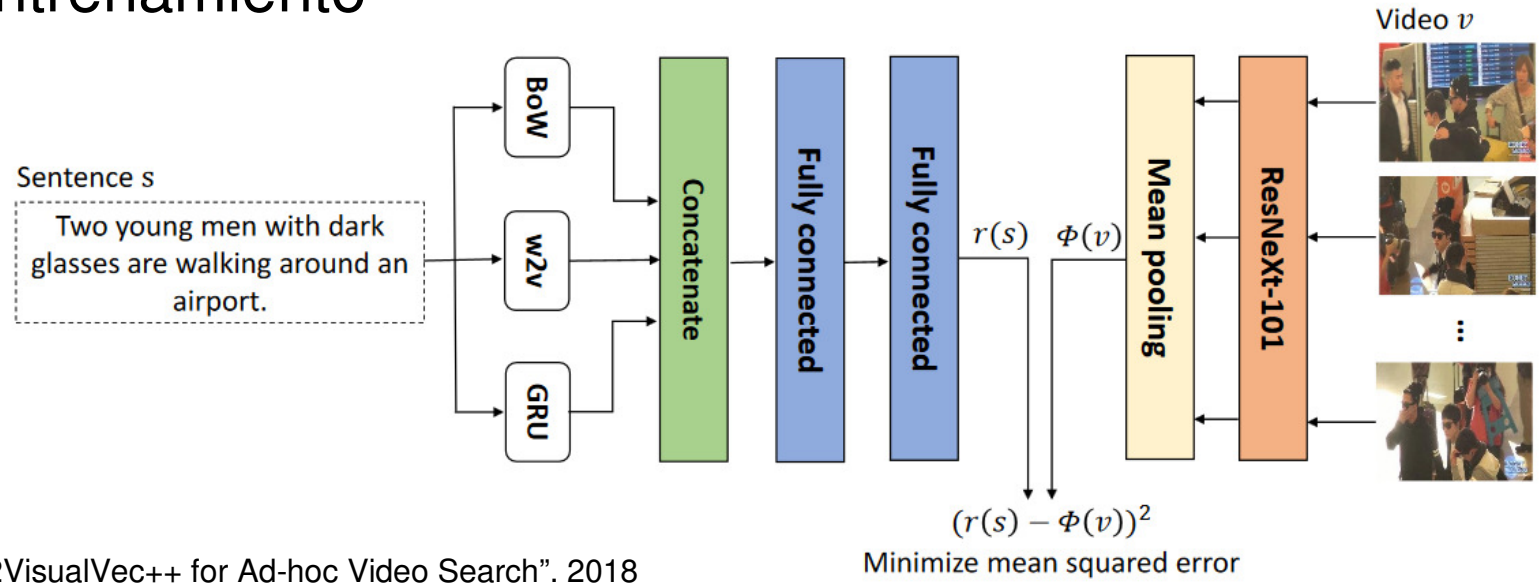
Dong, Li, Snoek. Predicting Visual Features from Text for Image and Video Caption Retrieval. 2018
https://github.com/danieljf24/w2vv

- Búsqueda de texto libre:
  - Calcular los vectores visuales de las imágenes del dataset
  - Calcular el vector textual de la consulta
  - Usar la red MLP (entrenada con COCO) y obtener su conversión a vector visual
  - Buscar los vectores visuales más cercanos en las imágenes del dataset



Word2VisualVec++

- En vez de convertir un descriptor en el otro, generar un espacio combinado al que ambos espacios se proyectan
- Se requiere una función de distancia que se desea minimizar
- Selector de los mejores pares para mejorar el entrenamiento

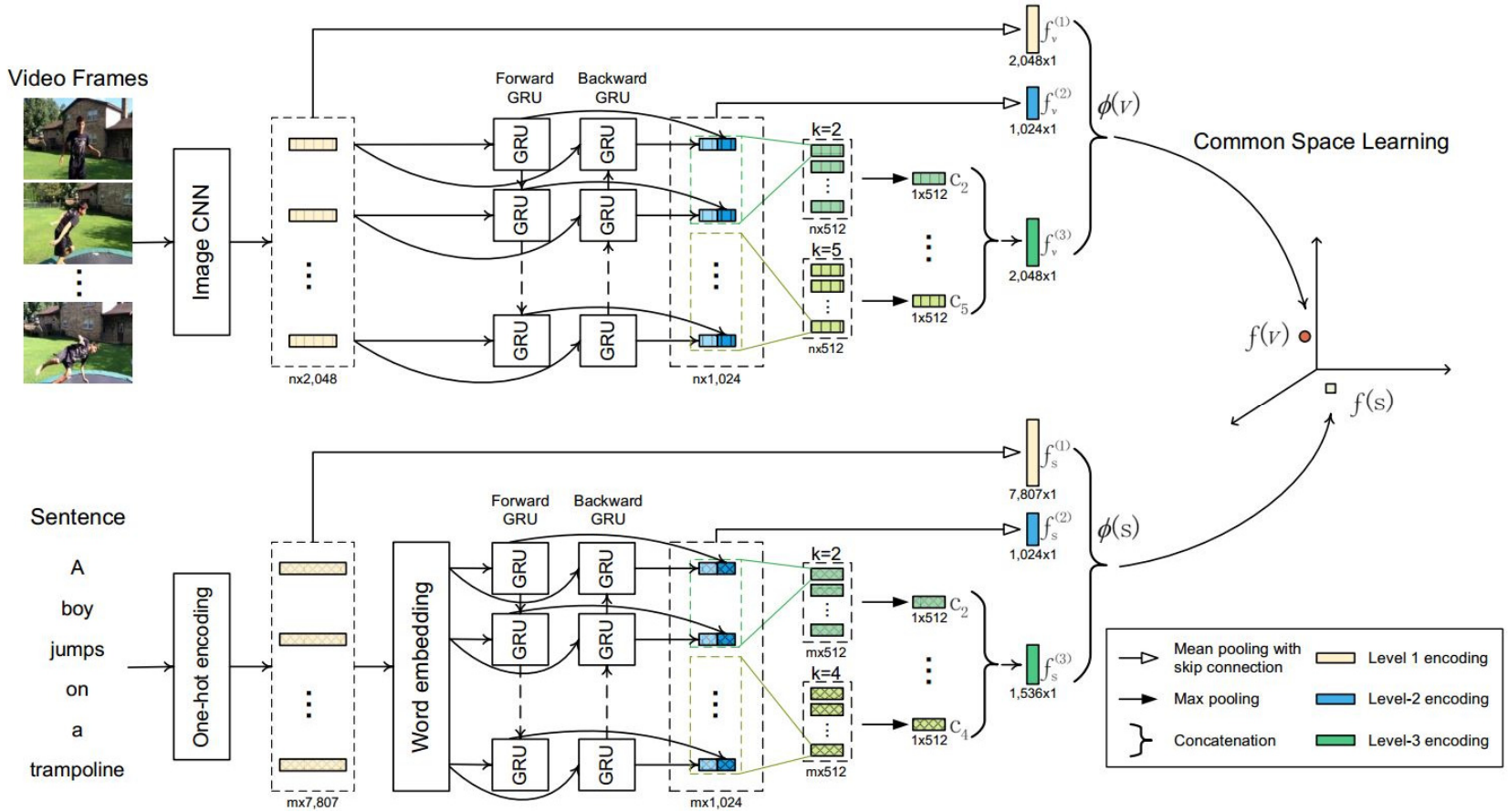


Li et al. "Word2VisualVec++ for Ad-hoc Video Search". 2018
https://www-nlpir.nist.gov/projects/tvpubs/tv18.slides/rucmm.avs.slides.pdf

Dual Encoding

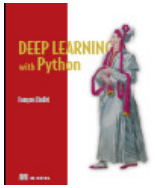
Dong et al. “Dual Encoding for Zero-Example Video Retrieval”. 2019
https://github.com/danieljf24/dual_encoding

- Combinar imagen y texto de forma simétrica:

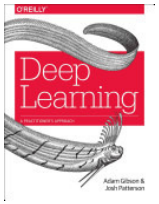


Bibliografía

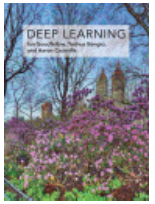
- **Deep Learning with Python**. Chollet. 2018.



- **Deep Learning: A Practitioner's Approach**. Patterson, Gibson. 2017.



- **Deep Learning**. Goodfellow, Bengio, Courville. 2016.



- Curso de Stanford (http://cs231n.github.io/)
  - http://cs231n.github.io/neural-networks-1/
  - http://cs231n.github.io/neural-networks-2/