Attention
&
Transformer Networks

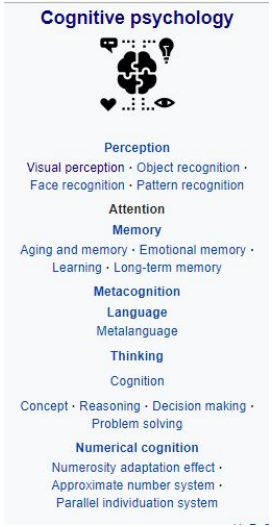
Παρασκευή Τζούβελη

AI|LS
ARTIFICIAL INTELLIGENCE
& LEARNING SYSTEMS LABORATORY

ε.δε.μ²
ΕΠΙΣΤΗΜΗ ΔΕΔΟΜΕΝΩΝ & ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

Attention: Μηχανισμός προσοχής

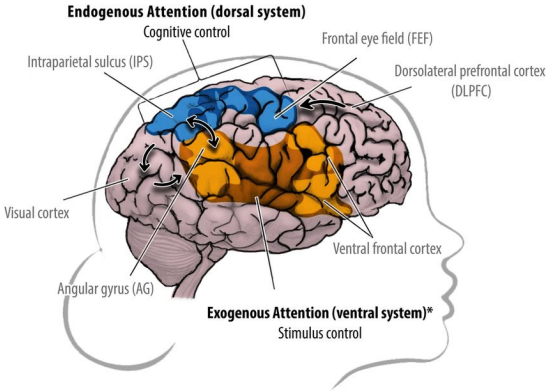
- Η προσοχή είναι η συμπεριφορική και γνωστική διαδικασία της επιλεκτικής συγκέντρωσης σε μια διακριτή πτυχή της πληροφορίας, είτε θεωρείται υποκειμενική είτε αντικειμενική, ενώ αγνοούνται άλλες αντιληπτές πληροφορίες.



- Frontiers | Attention in Psychology, Neuroscience, and Machine Learning
- Attention (Stanford Encyclopedia of Philosophy)

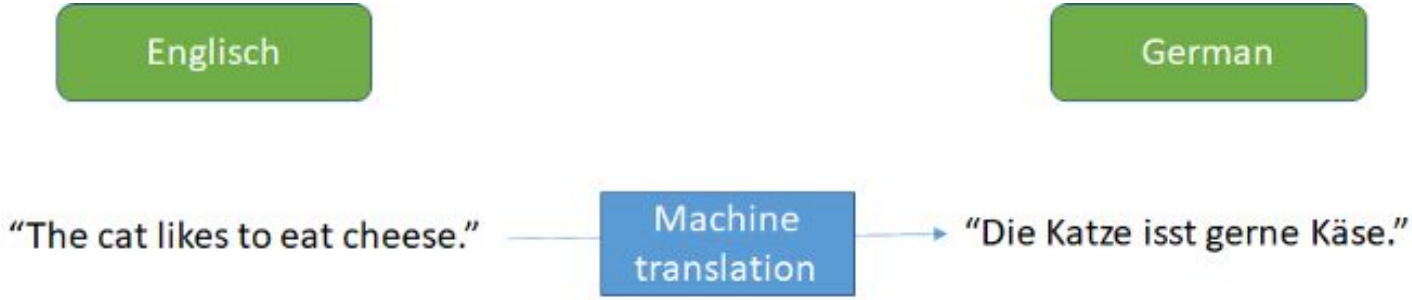
Μηχανισμός προσοχής : Attention

- ➔ Ο πυρήνα της αρχιτεκτονικής του Transformer
- ➔ Εμπνέεται από την προσοχή στον ανθρώπινο εγκέφαλο.
  - ◆ Φανταστείτε τον εαυτό σας να βρίσκεστε σε ένα πάρτι.
    - Μπορείτε να αναγνωρίσετε το όνομά σας που ακούγεται στην άλλη πλευρά του δωματίου, ακόμα κι αν χαθεί σε όλο τον άλλο θόρυβο.
      - Ο **εγκέφαλός** σας μπορεί να **επικεντρωθεί σε πράγματα** που θεωρεί **σημαντικά** και να φιλτράρει όλες τις περιττές πληροφορίες.



Endogenous (dorsal) versus exogenous (ventral) attention systems....

Attention: κίνητρο



➔ Μηχανική μετάφραση

- Γινόταν κυρίως με χρήση επαναλαμβανόμενων νευρωνικών δικτύων όπως το LSTM ή το GRU.
  - Δυσκολία στο να μάθουν εξαρτήσεις μεταξύ λέξεων, οι οποίες βρίσκονται πολύ μακριά σε μια πρόταση

Attention: διαισθητικά

Ένας μεταφραστής χωρίς Attention:

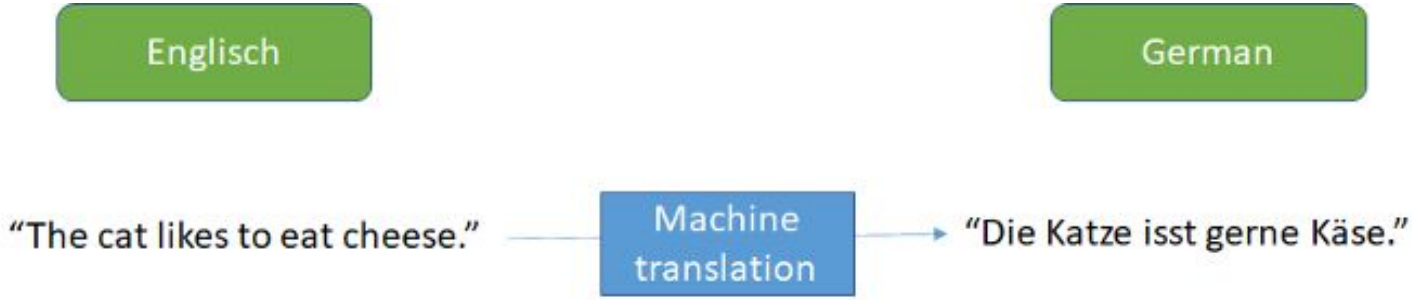
- διαβάζει το αγγλικό κείμενο από την αρχή έως το τέλος,
- μόλις τελειώσει, αρχίζει να μεταφράζει στα γερμανικά, λέξη προς λέξη.

Είναι πιθανό ότι εάν η πρόταση είναι πολύ μεγάλη, να έχει ξεχάσει τι έχει διαβάσει πριν

Ένας μεταφραστής με Attention:

- διαβάζει το αγγλικό κείμενο ενώ γράφει τις λέξεις-κλειδιά από την αρχή έως το τέλος,
- μετά αρχίζει να μεταφράζει στα γερμανικά, χρησιμοποιώντας τις λέξεις-κλειδιά που έχει γράψει.

Attention: κίνητρο



➔ Η μετάφραση λέξη προς λέξη δεν αποδίδει.

- Πρέπει με κάποιο τρόπο να τροφοδοτήσουμε πληροφορίες σχετικά με ολόκληρη την εισαγόμενη πρόταση στο μοντέλο αυτόματης μετάφρασης, ώστε να μπορεί να κατανοήσει το πλαίσιο των λέξεων.
  - Δεδομένου ότι τα περισσότερα μοντέλα αυτόματης μετάφρασης εξάγουν μία - μία λέξη τη φορά, πρέπει να δώσουμε στο μοντέλο επίσης πληροφορίες σχετικά με τα μέρη που έχει ήδη μεταφράσει.

➔ Οι **transformers** εισήχθησαν για να αντιμετωπίσουν αυτό το πρόβλημα αφαιρώντας την επανάληψη (recurrence) και αντικαθιστώντας την με έναν μηχανισμό προσοχής.

Transformers

Attention is All you Need, Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, NiPS 2017

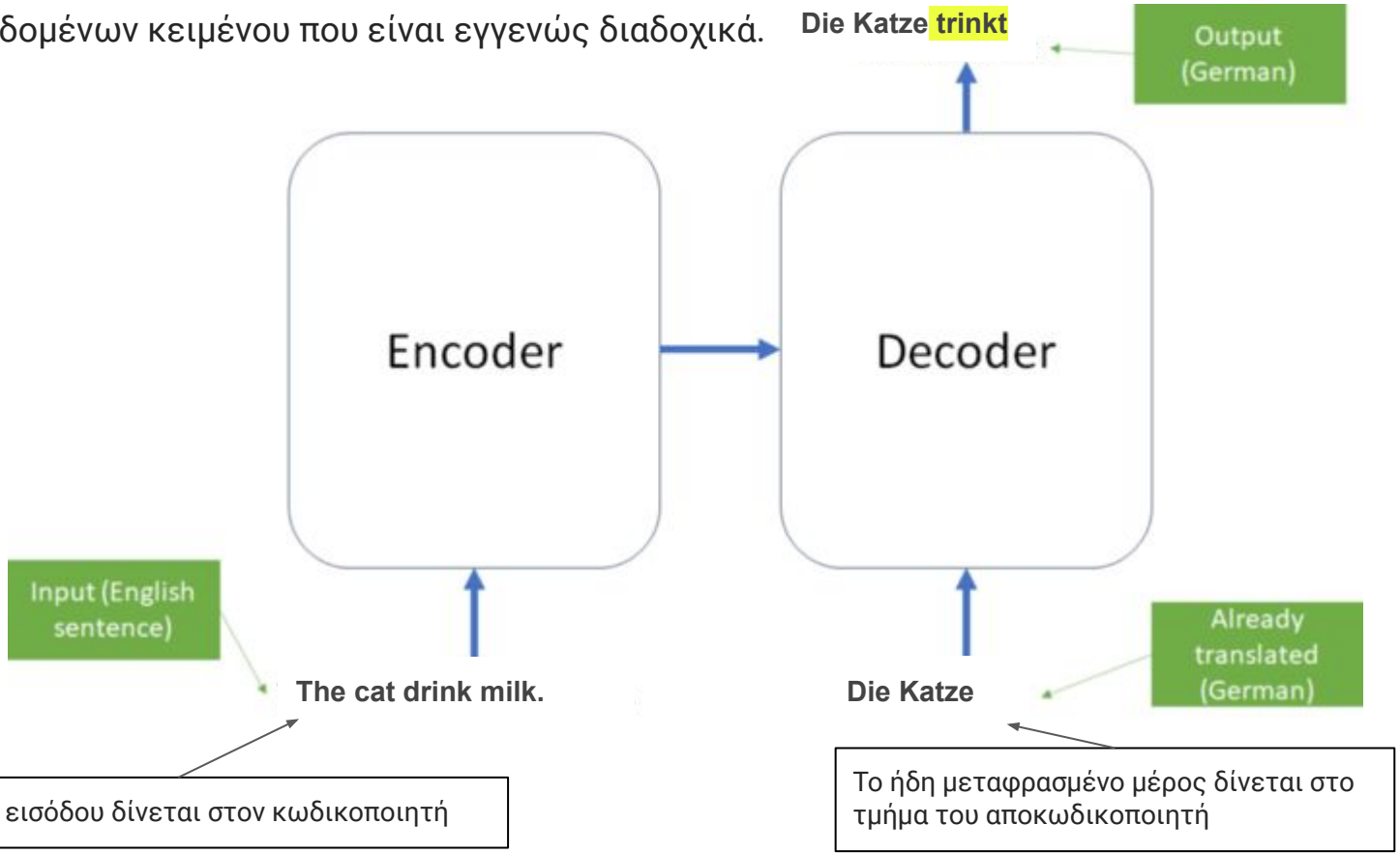
“The Transformer is the first transduction model relying entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution.”

“transduction”: μεταγωγή =μετατροπή των ακολουθιών εισόδου σε ακολουθίες

- ➔ χειρίζεται πλήρως τις εξαρτήσεις μεταξύ εισόδου και εξόδου με μηχανισμό attention και επανάληψη.
- ➔ χρησιμοποιεί το attention για να αυξήσει την ταχύτητα με την οποία αυτά τα μοντέλα μπορούν να εκπαιδευτούν.
- ➔ προσφέρεται για parallel processing
- ➔ το Google Colab χρησιμοποιεί το transformer μέσω του Cloud TPU

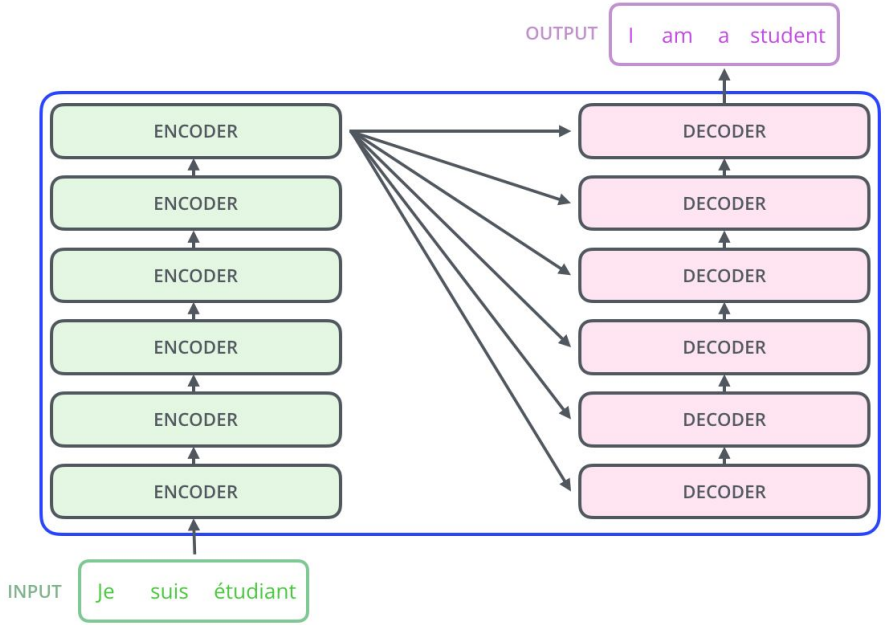
Αρχιτεκτονική Transformers

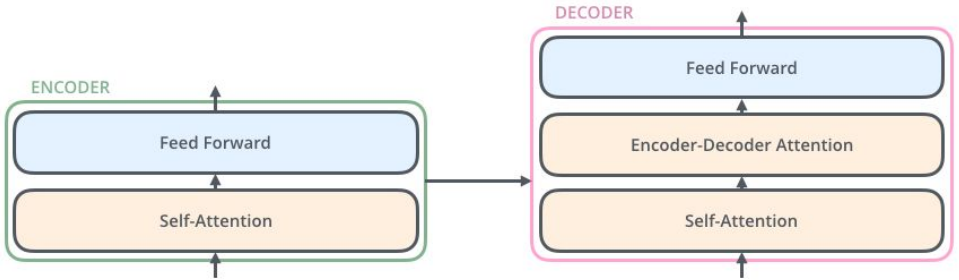
Χειρισμός δεδομένων κειμένου που είναι εγγενώς διαδοχικά.



Transformers

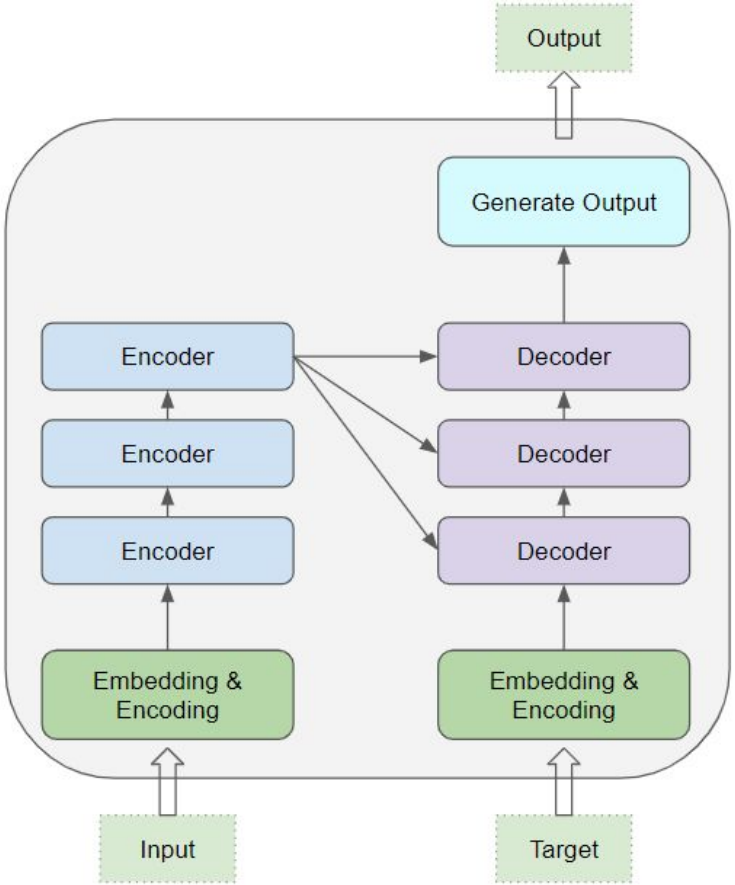
Η λειτουργία των Transformers βασίζεται σε αρχιτεκτονικές κωδικοποιητή - αποκωδικοποιητή (encoder - decoder) και σε attention μηχανισμούς.





http://jalammar.github.io/illustrated-transformer/

Αρχιτεκτονική Transformers

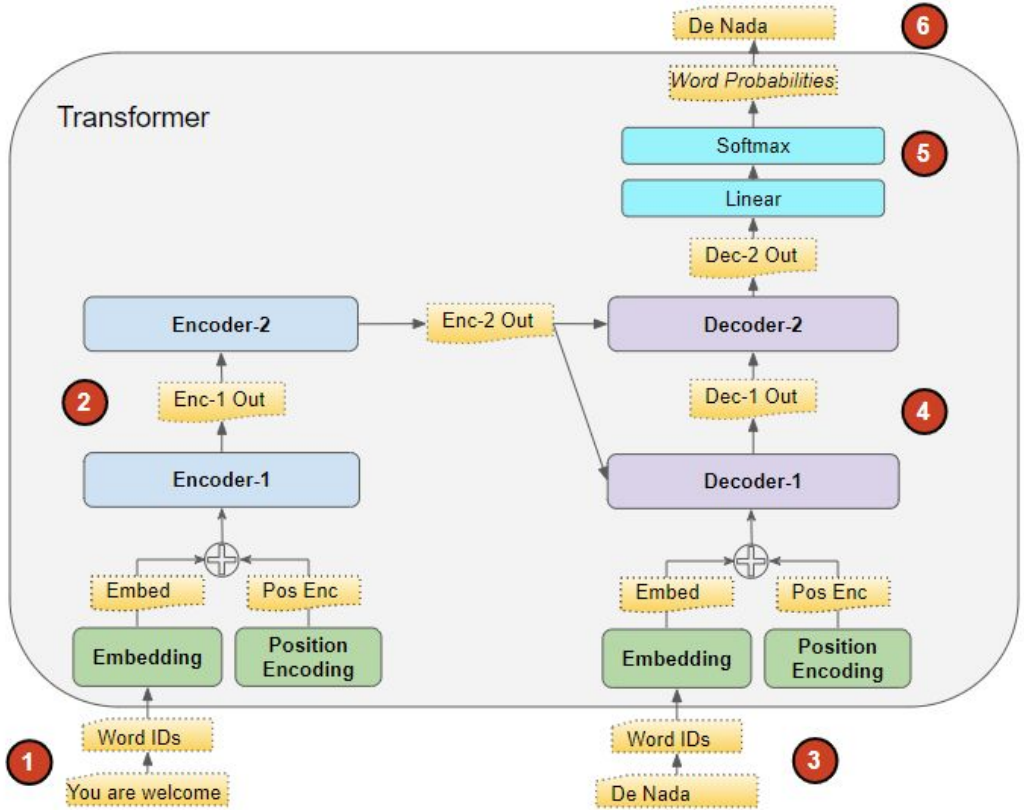


Στον πυρήνα του transformer, περιέχεται:

- Μια **στοίβα από επίπεδα κωδικοποιητή** και μία **στοίβα από επίπεδα αποκωδικοποιητή**.
  - Η στοίβα κωδικοποιητή (encoder) και η στοίβα αποκωδικοποιητή (decoder) έχουν τα αντίστοιχα **επίπεδα ενσωμάτωσης (embedding layers)** για τις αντίστοιχες εισόδους τους.
  - Όλοι οι κωδικοποιητές είναι πανομοιότυποι.
  - Όλοι οι αποκωδικοποιητές είναι πανομοιότυποι.
- Τέλος, υπάρχει ένα **επίπεδο εξόδου** για τη δημιουργία της τελικής εξόδου.

Transformers Explained Visually (Part 1): Overview of Functionality | by Ketan Doshi

Εκπαίδευση Transformer

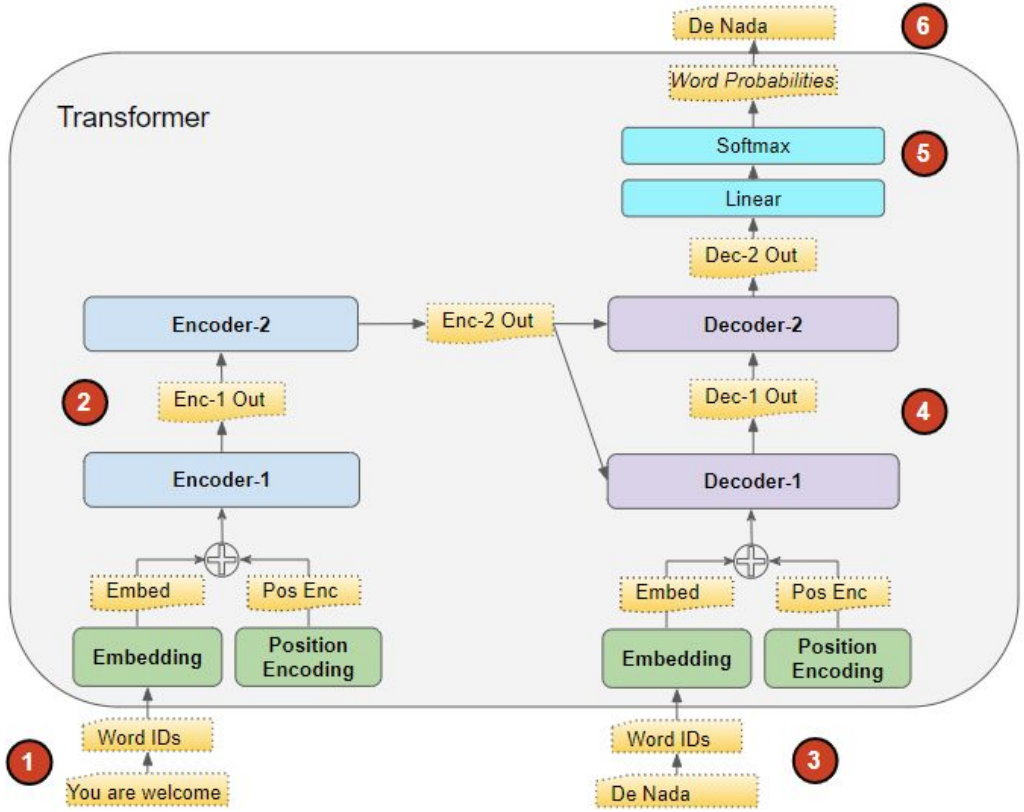


Τα δεδομένα εκπαίδευσης αποτελούνται από δύο μέρη:

- Η πηγή ή ακολουθία εισαγωγής (π.χ. "You are welcome" στα Αγγλικά, για ένα πρόβλημα μετάφρασης)
- Η αλληλουχία προορισμού ή στόχος (π.χ. "De nada" στα Ισπανικά)

Στόχος του transformer είναι να μάθει πώς να **εξαγάγει την ακολουθία στόχο**, χρησιμοποιώντας τόσο την ακολουθία εισόδου όσο και την ακολουθία στόχου.

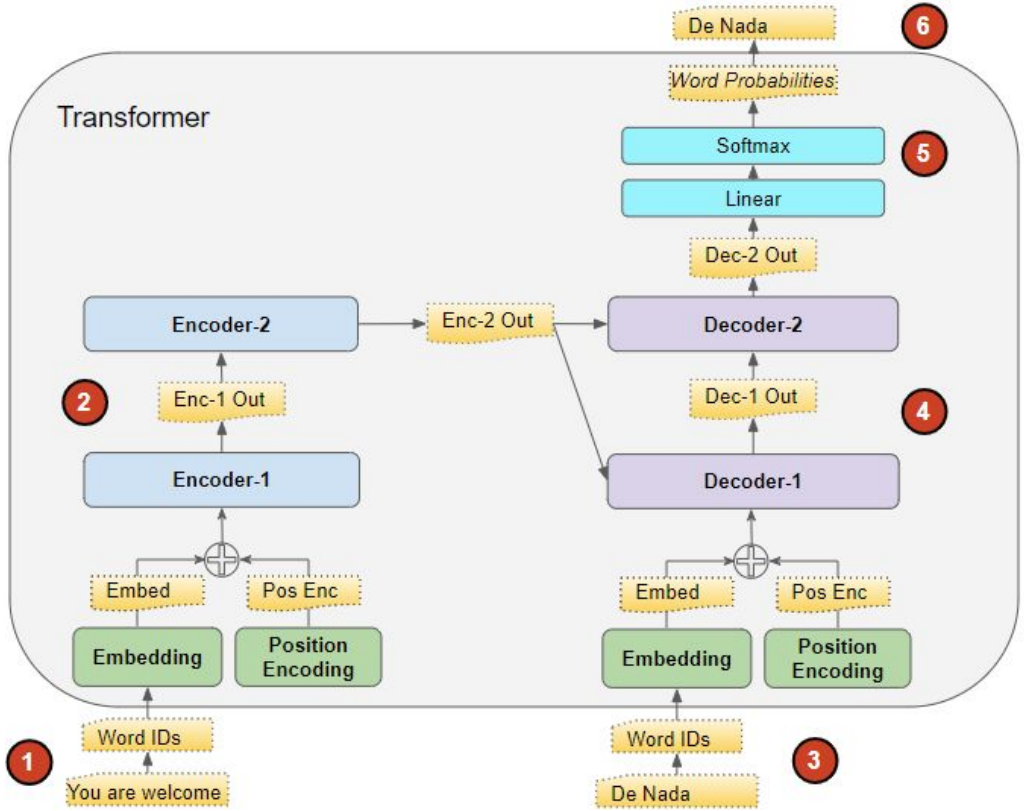
Εκπαίδευση Transformer



Ο transformer επεξεργάζεται τα δεδομένα ως εξής:

1. Η ακολουθία εισόδου μετατρέπεται σε embeddings (με κωδικοποίηση θέσης) και τροφοδοτείται στον κωδικοποιητή.
2. Η στοίβα των κωδικοποιητών το επεξεργάζεται και παράγει μια κωδικοποιημένη αναπαράσταση της ακολουθίας εισόδου.
3. Η ακολουθία στόχος προσαρτάται με ένα διακριτικό έναρξης πρότασης, μετατρέπεται σε embeddings (με κωδικοποίηση θέσης) και τροφοδοτείται στον αποκωδικοποιητή.

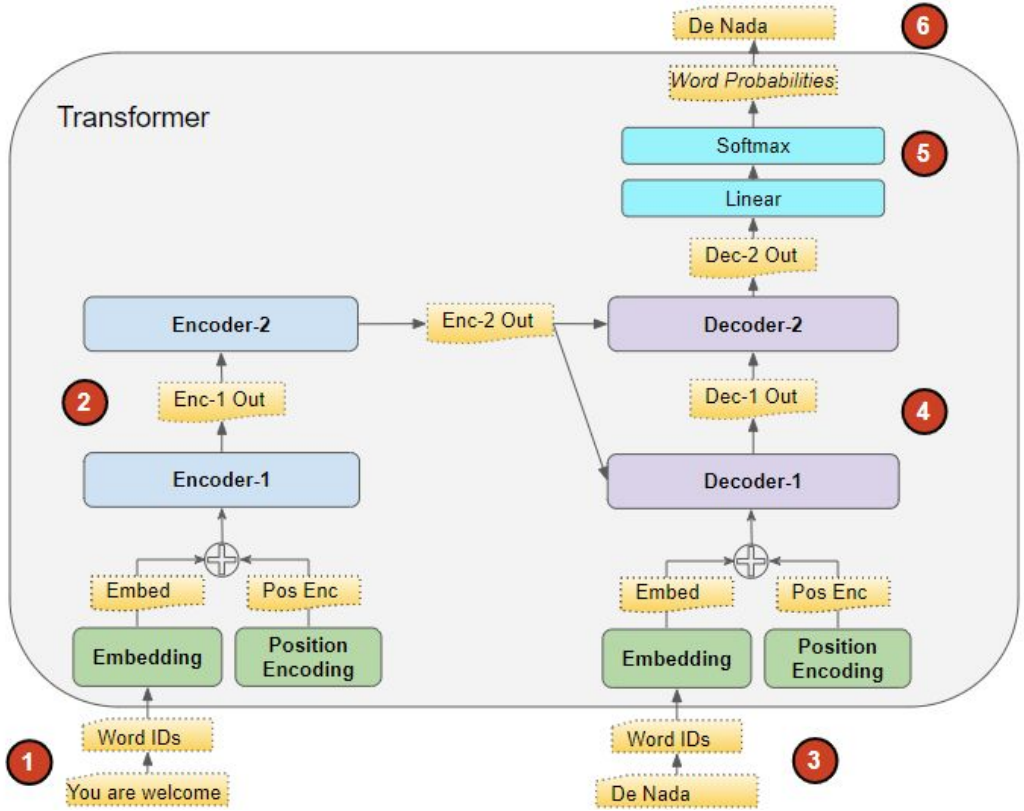
Εκπαίδευση Transformer



4. Η στοίβα των αποκωδικοποιητών το επεξεργάζεται αυτό μαζί με την κωδικοποιημένη αναπαράσταση της στοίβας κωδικοποιητή για να παράγει μια κωδικοποιημένη αναπαράσταση της ακολουθίας στόχου.

5. Το επίπεδο εξόδου το μετατρέπει σε πιθανότητες λέξης και στην τελική ακολουθία εξόδου.

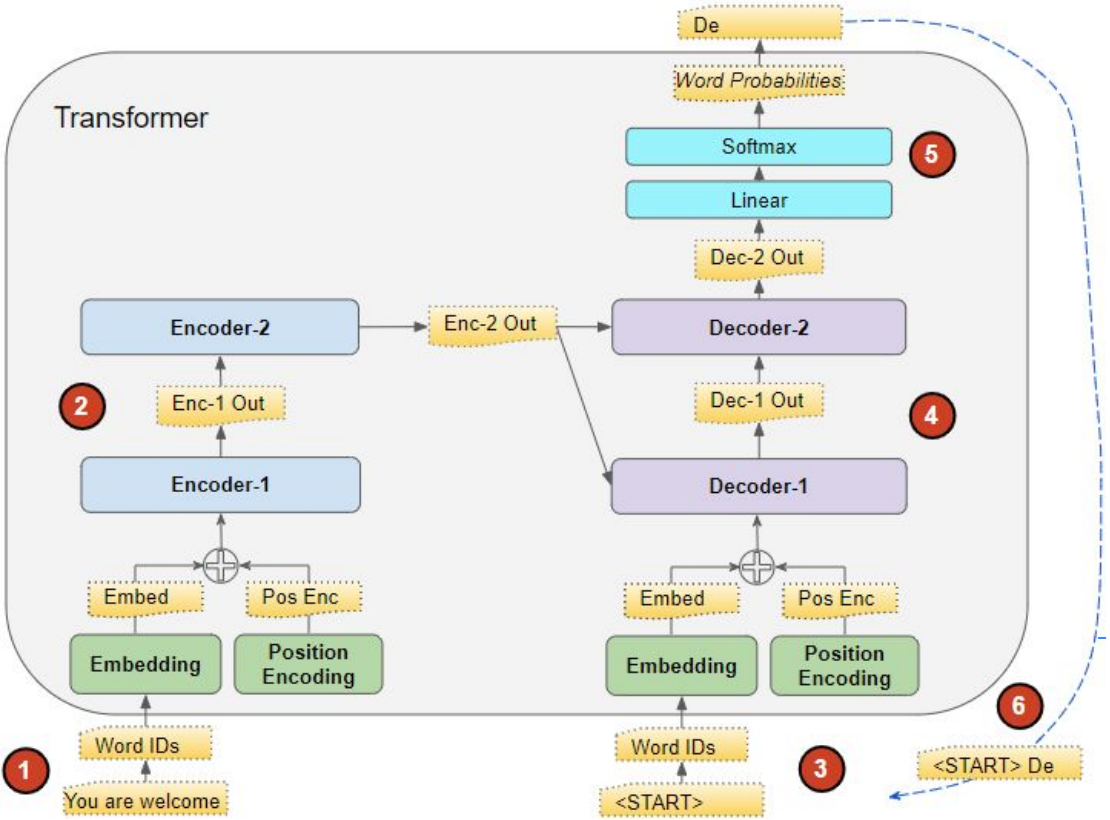
Εκπαίδευση Transformer



6. Η Loss function του Transformer συγκρίνει αυτήν την ακολουθία εξόδου με την ακολουθία στόχο από τα δεδομένα εκπαίδευσης.

   Αυτή η απώλεια χρησιμοποιείται για τη δημιουργία κλίσεων (gradients) για την εκπαίδευση του μετασχηματιστή κατά τη διάρκεια του back propagation.

Inference

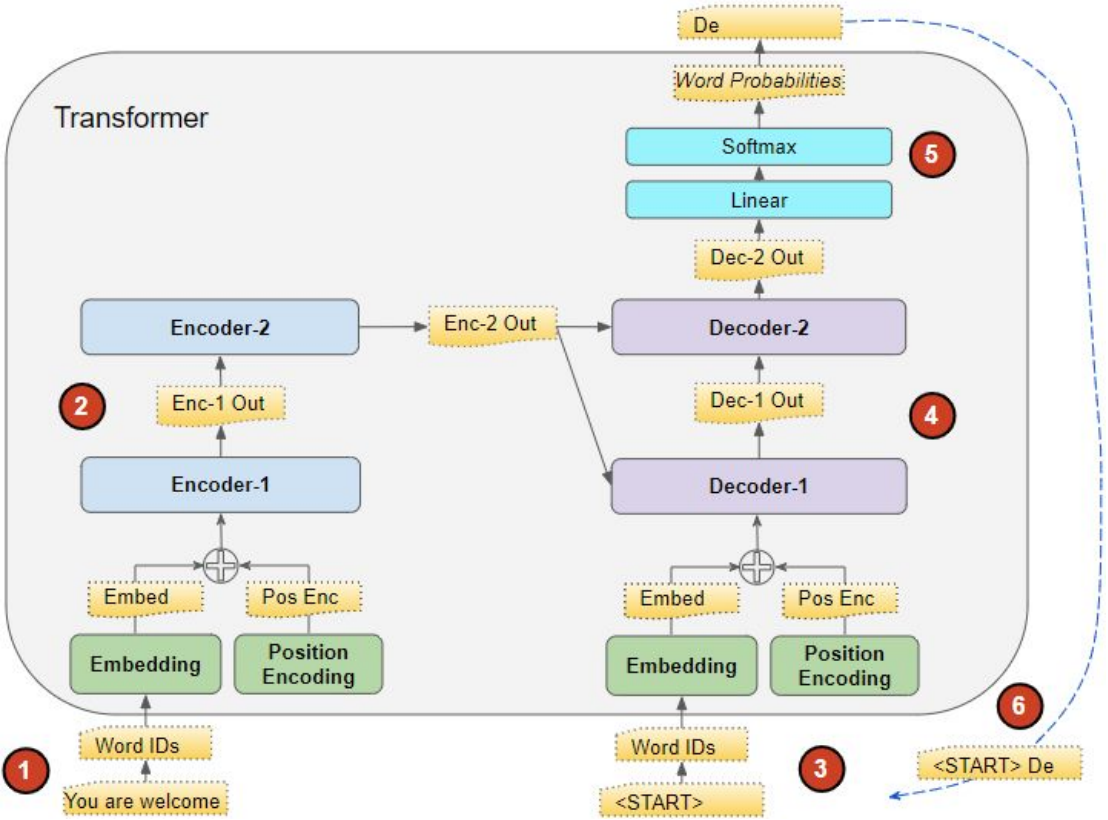


Κατά τη διάρκεια του Inference **έχουμε μόνο την ακολουθία εισόδου** και δεν έχουμε την ακολουθία-στόχο για να περάσει ως είσοδος στον αποκωδικοποιητή.

Ο **στόχος του transformer** είναι να **παράγει την ακολουθία στόχου μόνο από την ακολουθία εισόδου.**

* Σε κάθε χρονικό βήμα, τροφοδοτούμε εκ νέου ολόκληρη την ακολουθία εξόδου που έχει δημιουργηθεί μέχρι τώρα, και όχι μόνο την τελευταία λέξη.

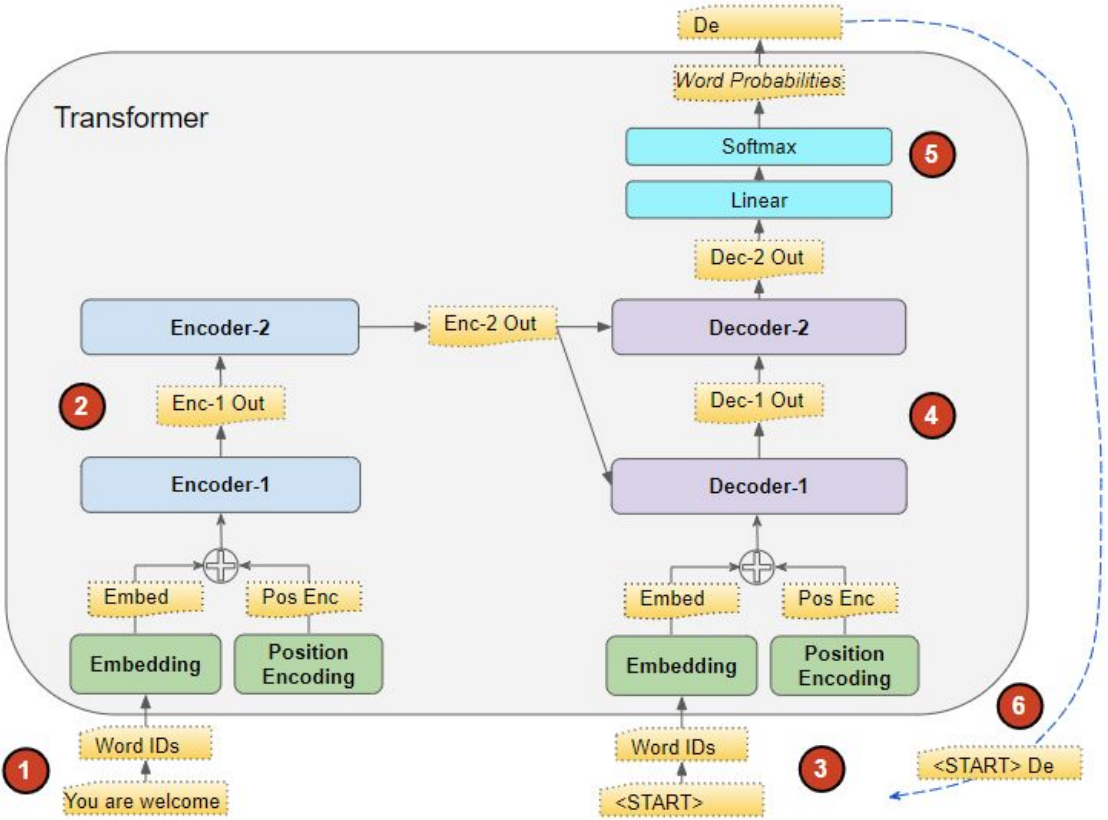
Inference



Η ροή δεδομένων κατά τη διάρκεια του Inference είναι:

1. Η ακολουθία εισόδου μετατρέπεται σε embeddings (με κωδικοποίηση θέσης) και τροφοδοτείται στον κωδικοποιητή.
2. Η στοίβα των κωδικοποιητών το επεξεργάζεται και παράγει μια κωδικοποιημένη αναπαράσταση της ακολουθίας εισόδου.

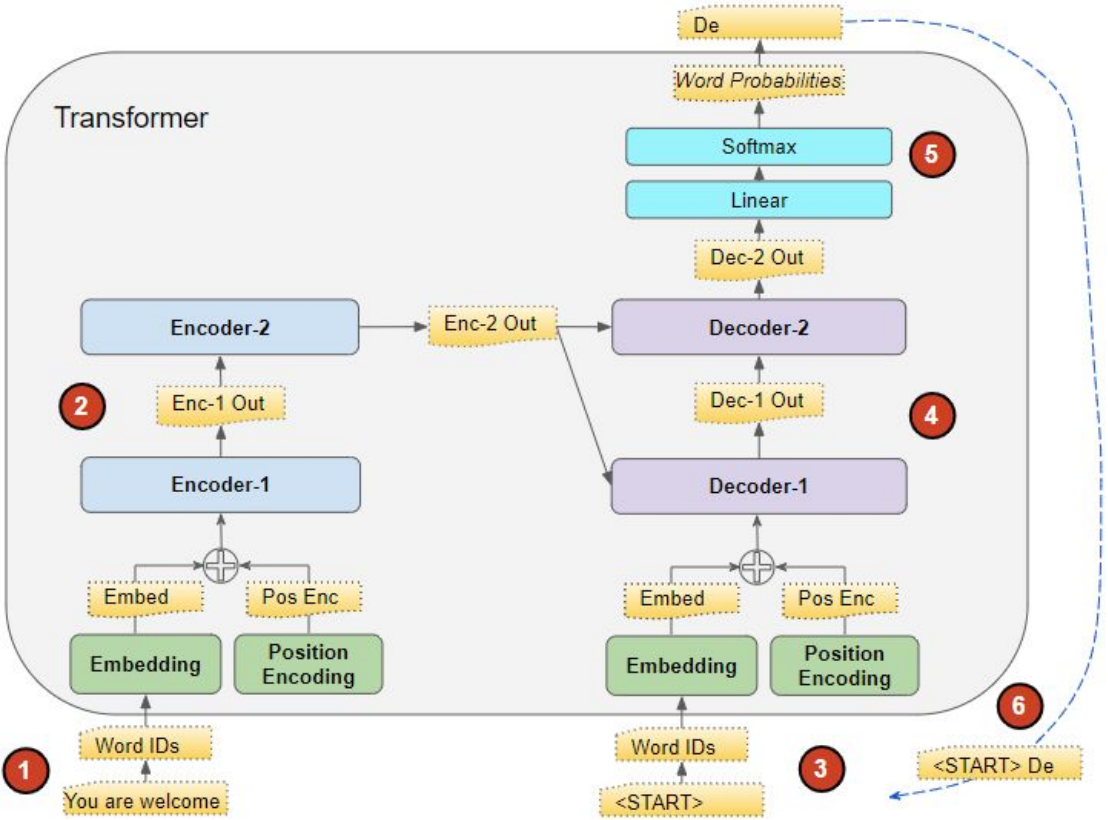
Inference



3. **Αντί για την ακολουθία στόχο**, χρησιμοποιούμε μια **κενή ακολουθία** με μόνο ένα διακριτικό αρχής πρότασης. Αυτό μετατρέπεται σε embeddings (με κωδικοποίηση θέσης) και τροφοδοτείται στον αποκωδικοποιητή.

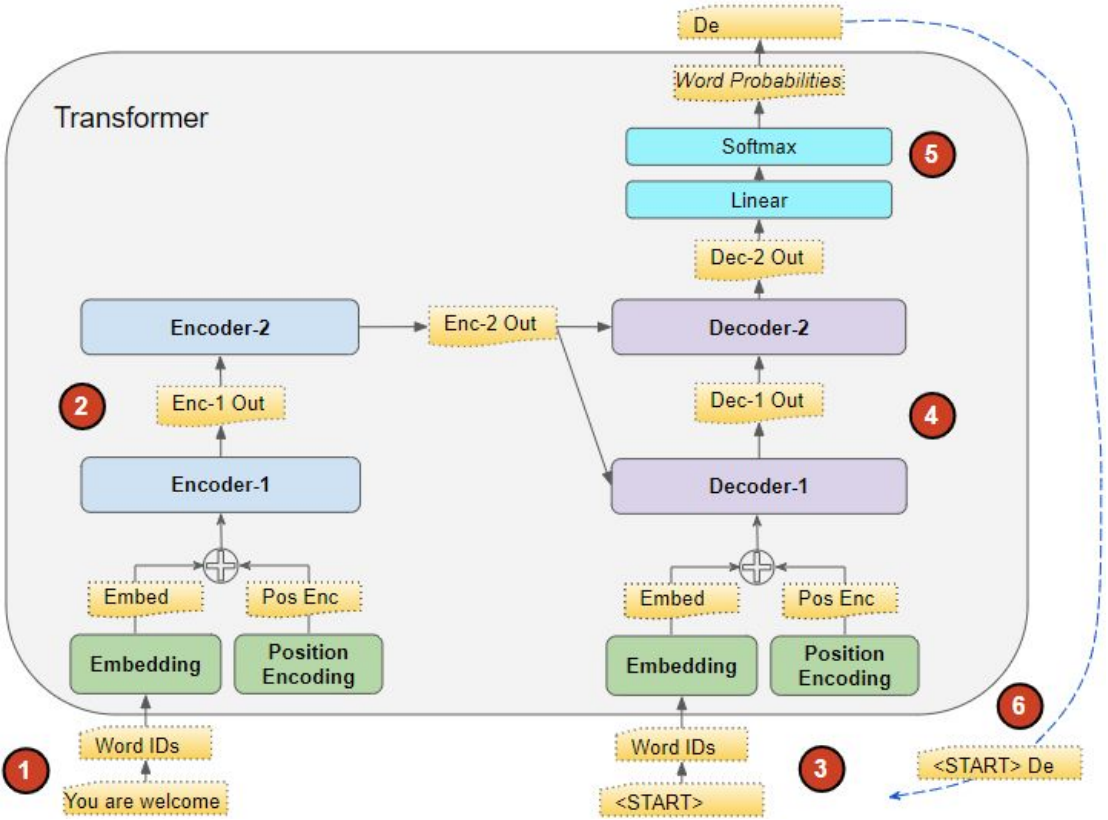
4. Η στοίβα των decoders το επεξεργάζεται αυτό μαζί με την κωδικοποιημένη αναπαράσταση της στοίβας encoders για να παράγει μια κωδικοποιημένη αναπαράσταση της ακολουθίας στόχου.

Inference



5. Το επίπεδο εξόδου το μετατρέπει σε πιθανότητες λέξεων και παράγει μια ακολουθία εξόδου.
   a. Λαμβάνουμε την τελευταία λέξη της ακολουθίας εξόδου ως την προβλεπόμενη λέξη.
   b. Αυτή η λέξη συμπληρώνεται τώρα στη δεύτερη θέση της ακολουθίας εισόδου του αποκωδικοποιητή μας, η οποία τώρα περιέχει ένα διακριτικό έναρξης πρότασης και την πρώτη λέξη.

Inference



6. Επιστρέψτε στο βήμα #3.

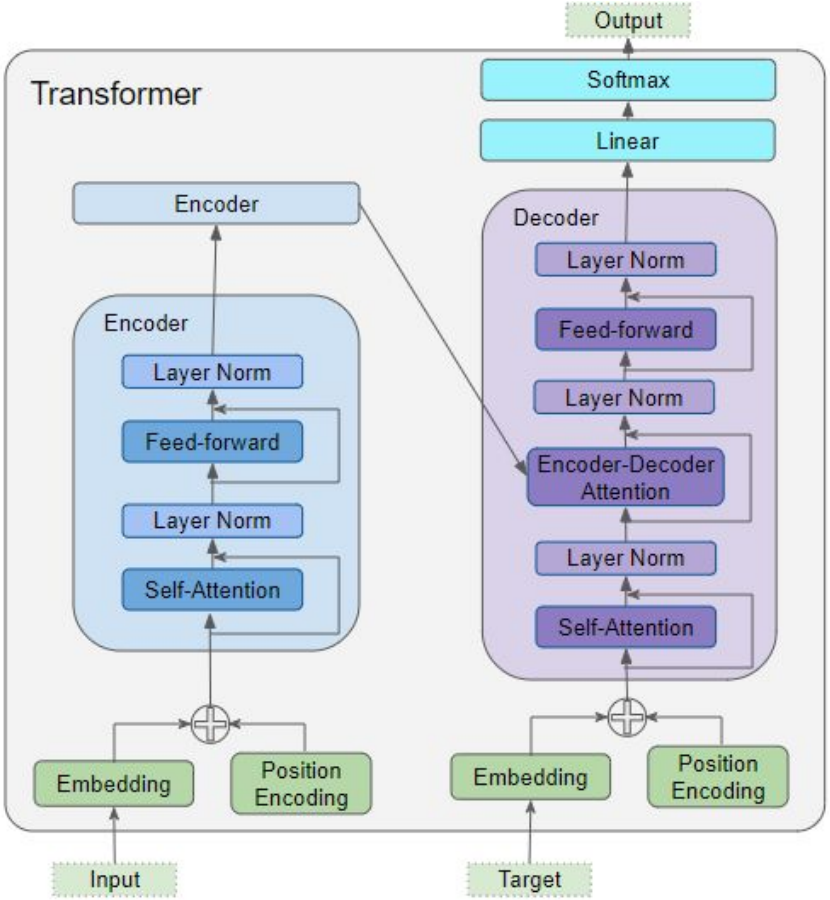
Όπως και πριν, τροφοδοτήστε τη νέα ακολουθία αποκωδικοποιητή στο μοντέλο.

Στη συνέχεια, πάρτε τη δεύτερη λέξη της εξόδου και προσθέστε την στην ακολουθία του αποκωδικοποιητή.

Επαναλάβετε αυτό μέχρι να προβλέψει ένα διακριτικό τέλους πρότασης.

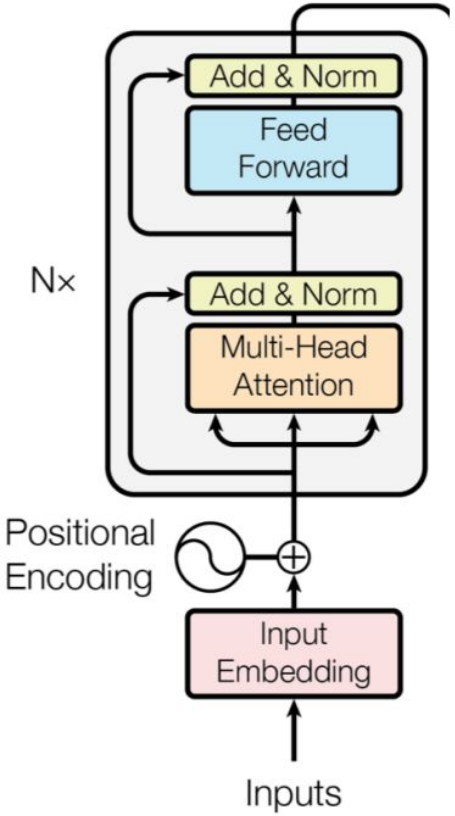
Σημειώστε ότι επειδή η ακολουθία του κωδικοποιητή δεν αλλάζει για κάθε επανάληψη, δεν χρειάζεται να επαναλαμβάνουμε τα βήματα #1 και #2 κάθε φορά

Transformers: look inside



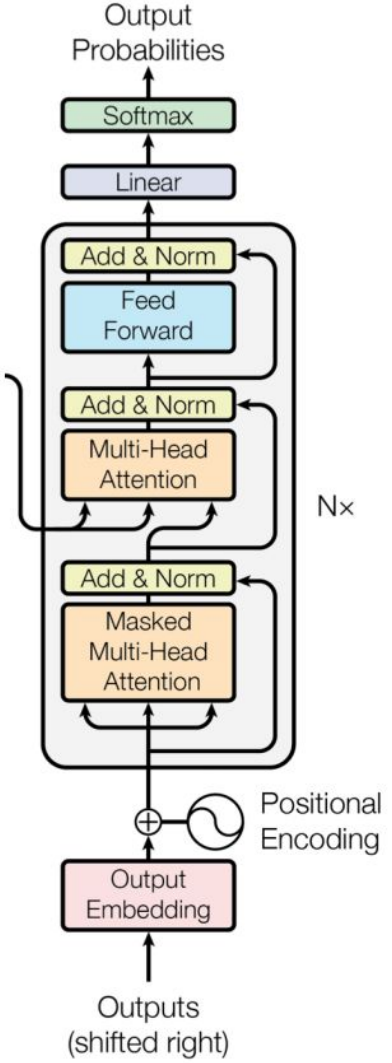
Encoder

- N = 6
- All layer output size 512
- Embedding
- Positional Encoding
- Multi-head Attention
- Residual Connection
- Position wise feed forward

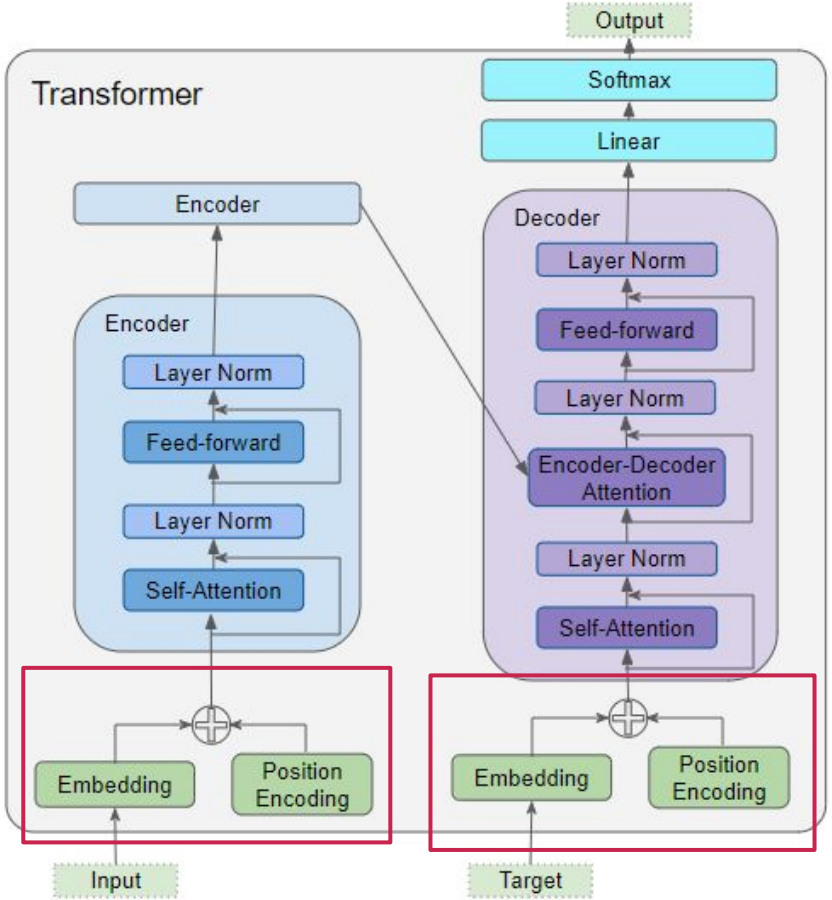


Decoder

- N = 6
- All layer output size 512
- Embedding
- Positional Encoding
- Residual Connection:
- LayerNorm(x + Sublayer(x))
- Multi-head Attention
- Position wise feed forward
- softmax: $$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$



Transformers : Embedding and Position Encoding



Όπως κάθε μοντέλο NLP, έτσι και ένας Transformer χρειάζεται δύο πράγματα για κάθε λέξη:

- τη σημασία της λέξης
- τη θέση της λέξης στην πρόταση

➔ Το επίπεδο ενσωμάτωσης (**Embedding layer**) **κωδικοποιεί τη σημασία της λέξης.**

➔ Το επίπεδο κωδικοποίησης θέσης (**Position Encoding layer**) **αντιπροσωπεύει τη θέση της λέξης**.

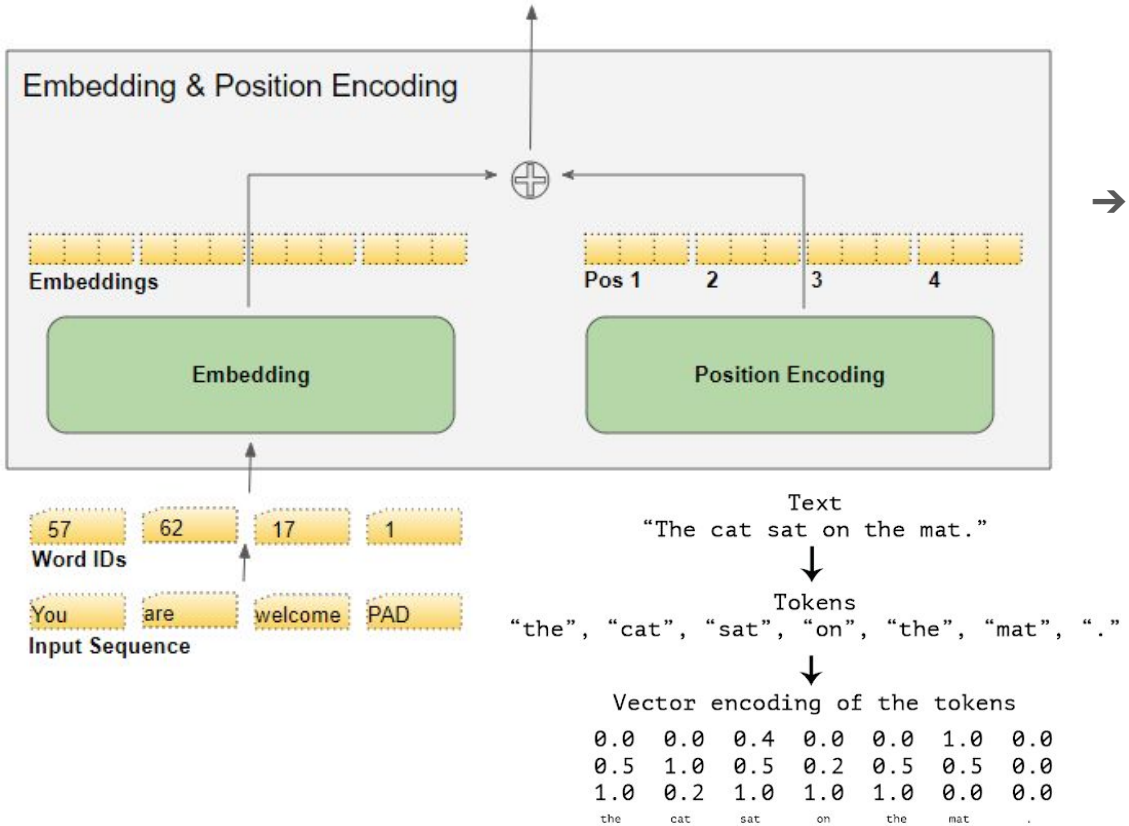
Ο Transformer συνδυάζει αυτές τις δύο κωδικοποιήσεις προσθέτοντάς τις.

Transformers : Embedding and Position Encoding

Ο Transformer έχει δύο στρώματα ενσωμάτωσης.

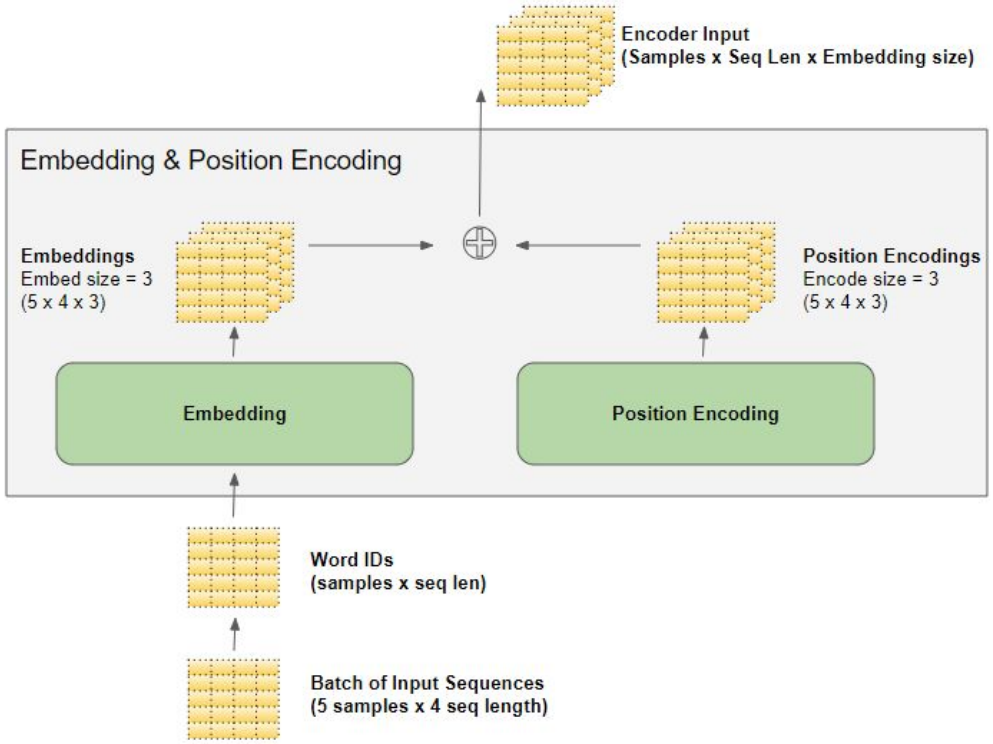
- ➔ Η ακολουθία εισόδου τροφοδοτείται στο embedding layer εισόδου.
- ➔ Η ακολουθία στόχου τροφοδοτείται στο embedding layer εξόδου αφού μετατοπιστούν οι στόχοι δεξιά κατά μία θέση και εισαγάγετε ένα διακριτικό έναρξης στην πρώτη θέση.

Transformers : Embedding



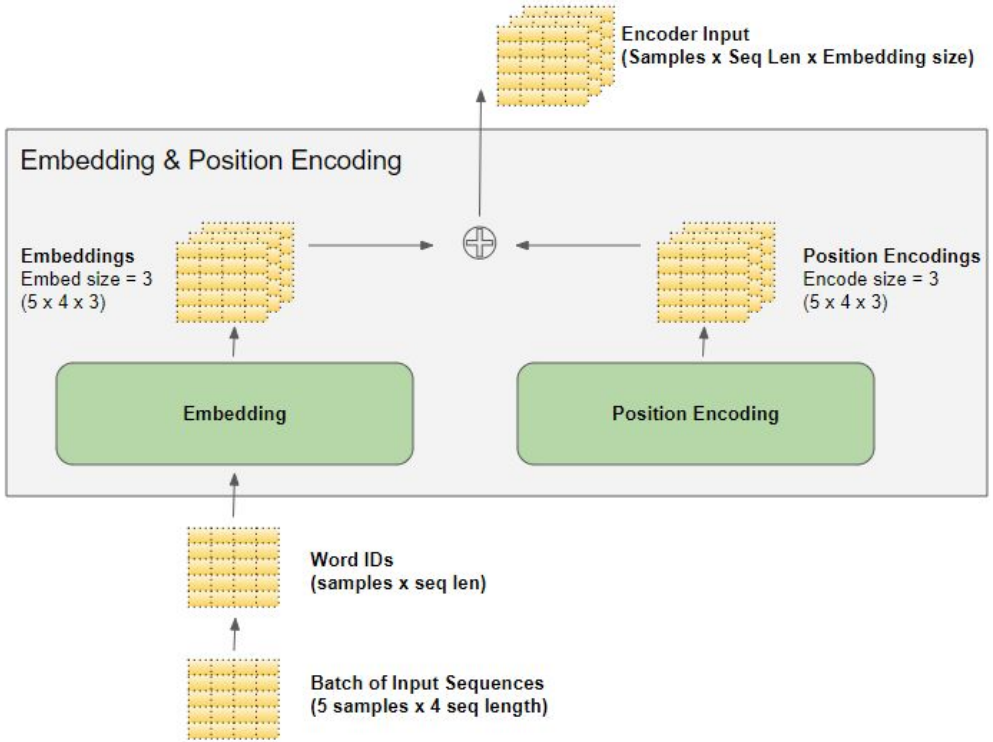
- Η ακολουθία εισόδου αντιστοιχίζεται σε αναγνωριστικά αριθμητικών λέξεων, χρησιμοποιώντας το λεξιλόγιό μας.
- Το Input Embedding and Position Encoding, του encoder και του decoder τροφοδοτείται με τα word IDs και παράγει μια **κωδικοποιημένη αναπαράσταση για κάθε λέξη** στην ακολουθία εισόδου, που καταγράφει το **νόημα και τη θέση κάθε λέξης**.

Transformers : Embedding



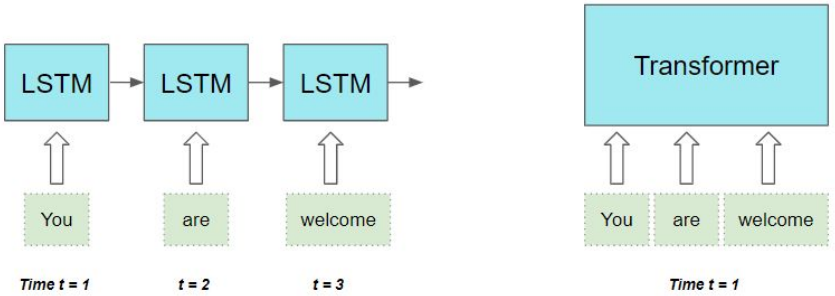
- Τα μοντέλα βαθιάς μάθησης επεξεργάζονται samples batch κάθε φορά.
- Τα επίπεδα Embedding and Position Encoding λειτουργούν με πίνακες που αντιπροσωπεύουν ένα sequence samples batch.
- Το Embedding λαμβάνει μια μήτρα σε σχήμα (δείγματα , μήκος ακολουθίας) των word IDs .
- Κωδικοποιεί κάθε word IDs σε ένα διάνυσμα λέξης του οποίου το μήκος είναι ίσο με το μέγεθος ενσωμάτωσης, με αποτέλεσμα μια μήτρα εξόδου σε σχήμα (δείγματα, μήκος ακολουθίας, μέγεθος ενσωμάτωσης).

Transformers : Embedding



- Η κωδικοποίηση θέσης χρησιμοποιεί ένα μέγεθος κωδικοποίησης που είναι ίσο με το μέγεθος του Embedding.
- Έτσι, παράγει μια μήτρα παρόμοιου σχήματος που μπορεί να προστεθεί στη μήτρα Embedding.
- Το σχήμα (δείγματα, μήκος ακολουθίας, μέγεθος embeddings) που παράγεται από τα επίπεδα Embedding and Position Encoding διατηρείται σε όλο τον transformer, καθώς τα δεδομένα ρέουν μέσω των Στοίβων Κωδικοποιητή και Αποκωδικοποιητή μέχρι να αναδιαμορφωθούν από τα τελικά επίπεδα εξόδου.

Transformers : Position Encoding



- Ένα LSTM(ή RNN ή GRU) υλοποιεί ένα βρόχο όπου κάθε λέξη εισάγεται διαδοχικά, οπότε γνωρίζει σιωπηρά τη θέση κάθε λέξης
- Οι transformers δεν χρησιμοποιούν RNN και όλες οι λέξεις σε μια ακολουθία εισάγονται παράλληλα.
- Οι πληροφορίες θέσης χάνονται και πρέπει να προστεθούν ξανά ξεχωριστά.

➔ Η κωδικοποίηση θέσης υπολογίζεται ανεξάρτητα από την ακολουθία εισόδου.

- Περιέχει σταθερές τιμές που εξαρτώνται μόνο από το μέγιστο μήκος της ακολουθίας.
- Για παράδειγμα, το πρώτο στοιχείο είναι ένας σταθερός κωδικός που υποδεικνύει την πρώτη θέση, το δεύτερο στοιχείο είναι ένας σταθερός κωδικός που υποδεικνύει τη δεύτερη θέση και ούτω καθεξής.

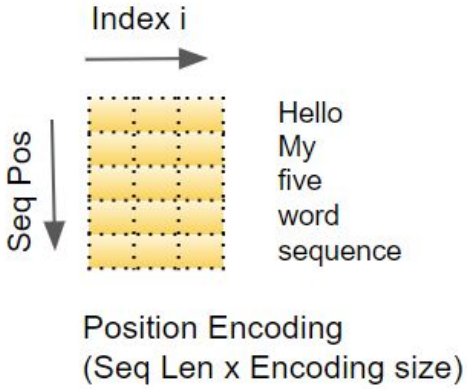
Transformers : Position Encoding

Αυτές οι σταθερές τιμές για τη κωδικοποίηση θέσης υπολογίζονται χρησιμοποιώντας τον τύπο:

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

όπου:

- pos είναι η θέση της λέξης στην ακολουθία
- d_model είναι το μήκος του διανύσματος κωδικοποίησης (ίδιο με το embedding διάνυσμα)
- i είναι η τιμή δείκτη σε αυτό το διάνυσμα.



Transformers : Position Encoding

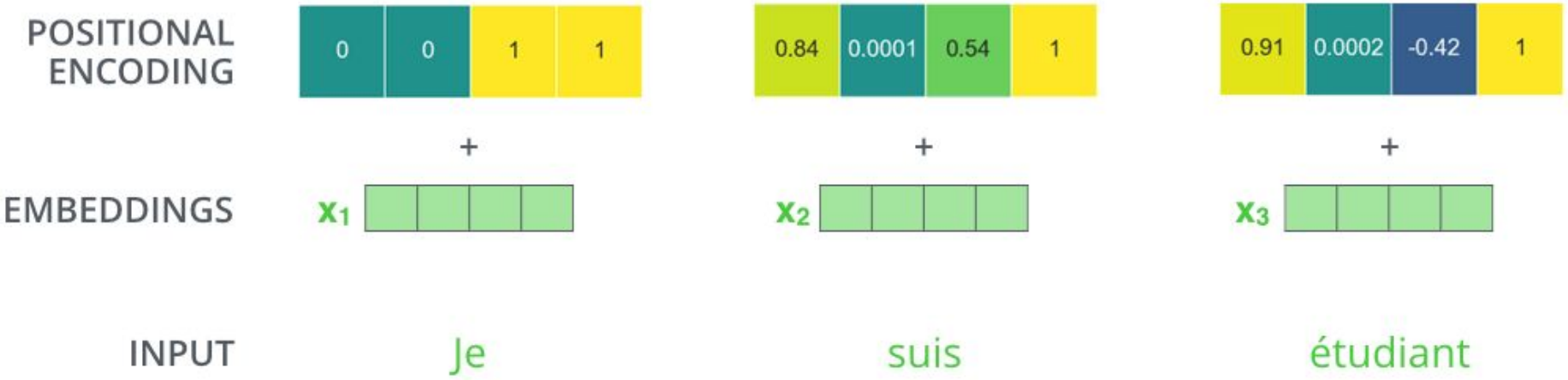
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

- Η ημιτονοειδής συνάρτηση λέει στο μοντέλο να δώσει προσοχή σε ένα συγκεκριμένο μήκος κύματος λ.
- Τα μήκη κύματος σχηματίζουν μια γεωμετρική πρόοδο από 2π έως 10000 · 2π
- Επιλέχθηκε η ημιτονοειδή συνάρτηση γιατί μπορεί να επιτρέψει στο μοντέλο να προεκταθεί σε μήκη ακολουθίας μεγαλύτερα από αυτά που συναντώνται κατά την εκπαίδευση.

<u>Αναλυτικότερα:</u> Δίνεται ένα σήμα y(x) = sin (kx)

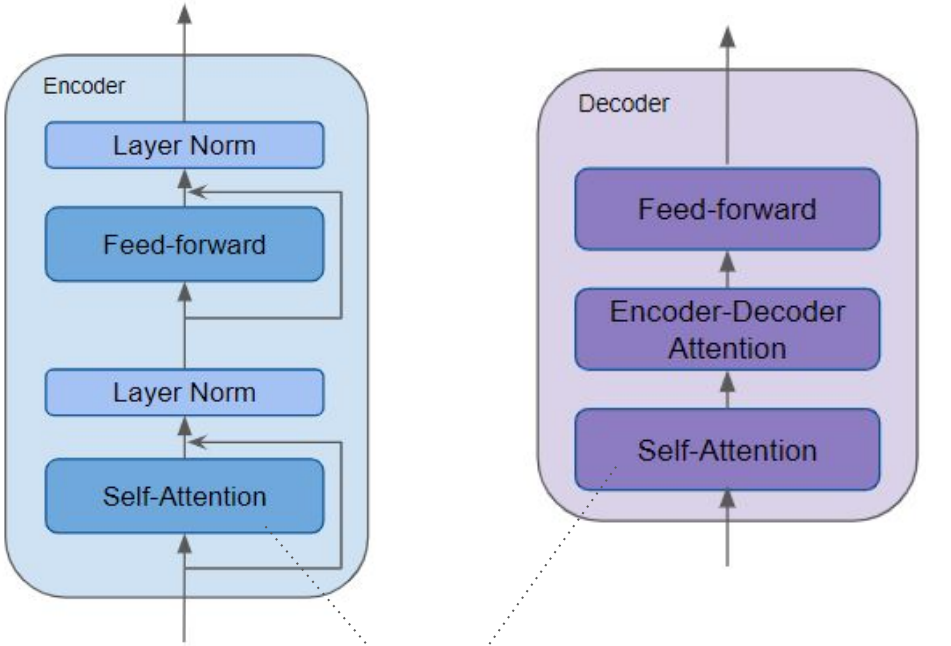
- Το μήκος κύματος θα είναι k = 2π/λ
- Στην περίπτωσή μας το λ θα εξαρτάται από τη θέση στην πρόταση.
- Το i χρησιμοποιείται για τη διάκριση μεταξύ περιττών και ζυγών θέσεων.
- Το d_{model}(=512), είναι η διάσταση των embedding διανυσμάτων.

Transformers : Position Encoding



Ένα παράδειγμα κωδικοποίησης θέσης με μέγεθος embedding ίσο με 4 (d_{model}=4)

Αρχιτεκτονική Transformers



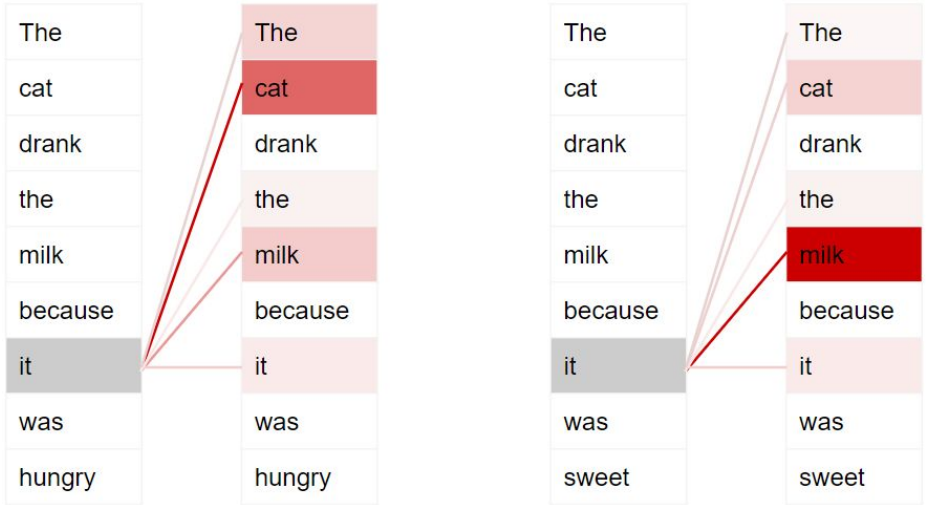
Κάθε κωδικοποιητής και αποκωδικοποιητής έχει το δικό του σύνολο βαρών.

Υπολογίζει τη σχέση μεταξύ διαφορετικών λέξεων στην ακολουθία

Αρχιτεκτονική Transformers: Self attention

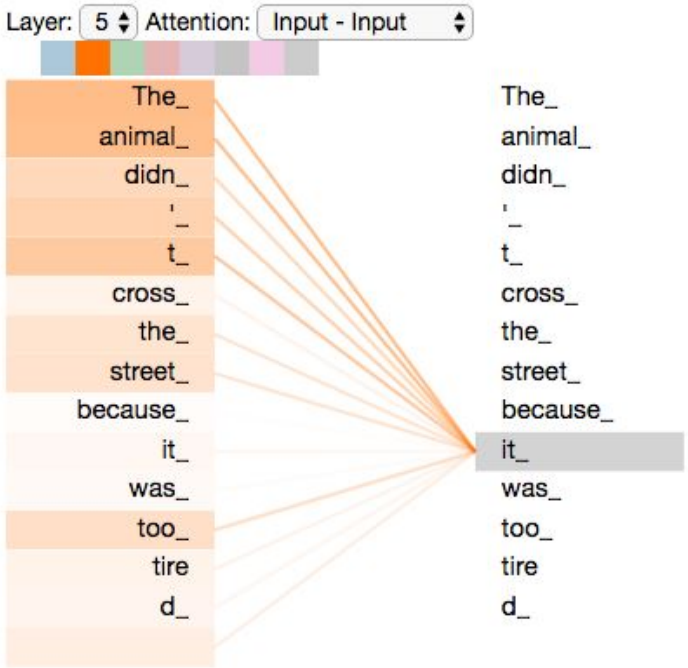
Παράδειγμα: Ας θεωρήσουμε δύο προτάσεις:

- The cat drank the milk because **it** was hungry.→ το 'it' αναφέρεται στο 'cat'
- The cat drank the milk because **it** was sweet.→ το 'it' αναφέρεται στο 'milk'



➔ Όταν το μοντέλο επεξεργάζεται τη λέξη «it», το **self-attention** δίνει στο μοντέλο περισσότερες πληροφορίες σχετικά με τη σημασία της, ώστε να μπορεί να συσχετίσει το «it» με τη σωστή λέξη.

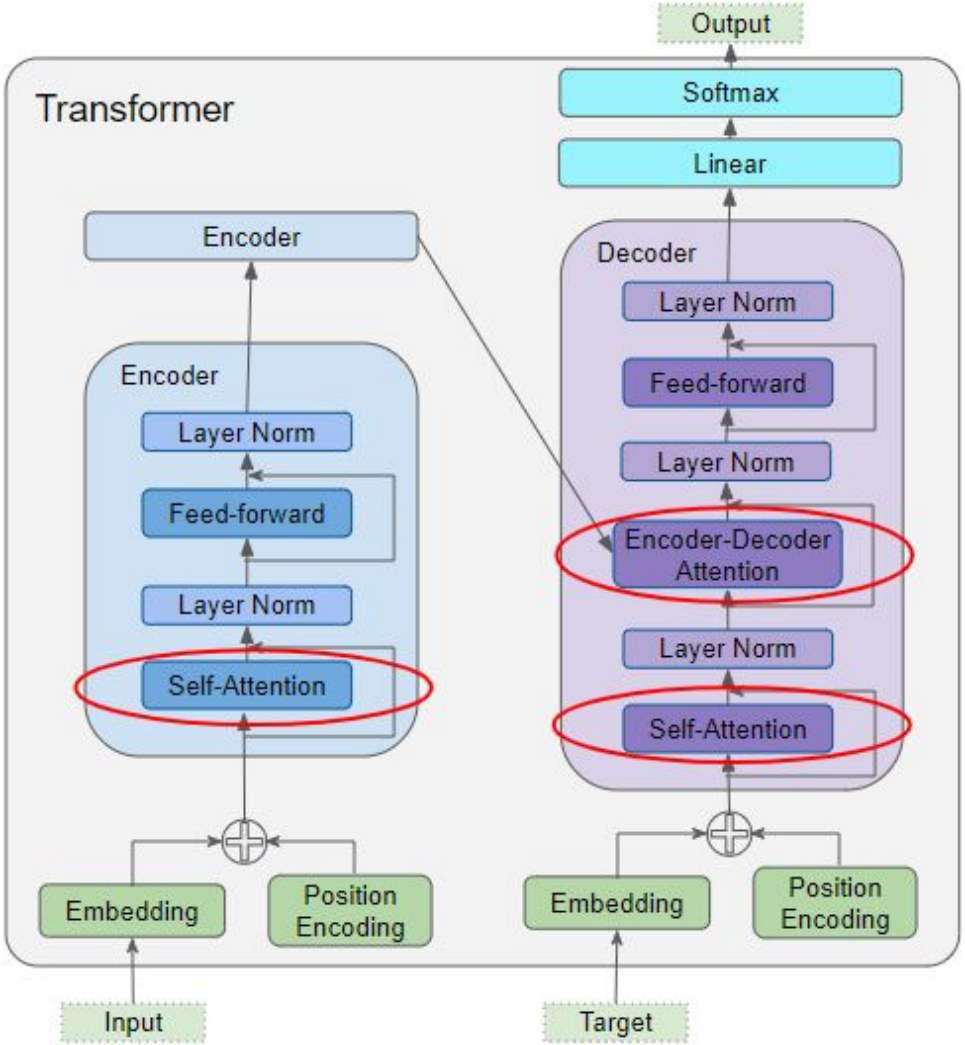
Αρχιτεκτονική Transformers: Self attention



"The animal didn't cross the street because it was too tired"

- Το self-attention ενσωματώνει την «κατανόηση» άλλων σχετικών λέξεων μέσω αυτής που επεξεργαζόμαστε αυτήν τη στιγμή.
- Μπορεί να εξετάσει άλλες θέσεις στην ακολουθία εισαγωγής για ενδείξεις που μπορούν να οδηγήσουν σε καλύτερη κωδικοποίηση της λέξη που πάει να κωδικοποιήσει

(σκεφτείτε πώς η διατήρηση μιας κρυφής κατάστασης επιτρέπει σε ένα RNN να ενσωματώσει την αναπαράσταση των προηγούμενων λέξεων / διανυσμάτων που έχει επεξεργαστεί με την τρέχουσα που επεξεργάζεται)

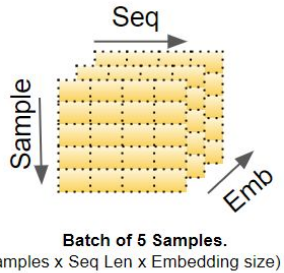


Self Attention

Το επίπεδο Self attention λαμβάνει την είσοδό του με τη μορφή τριών παραμέτρων:

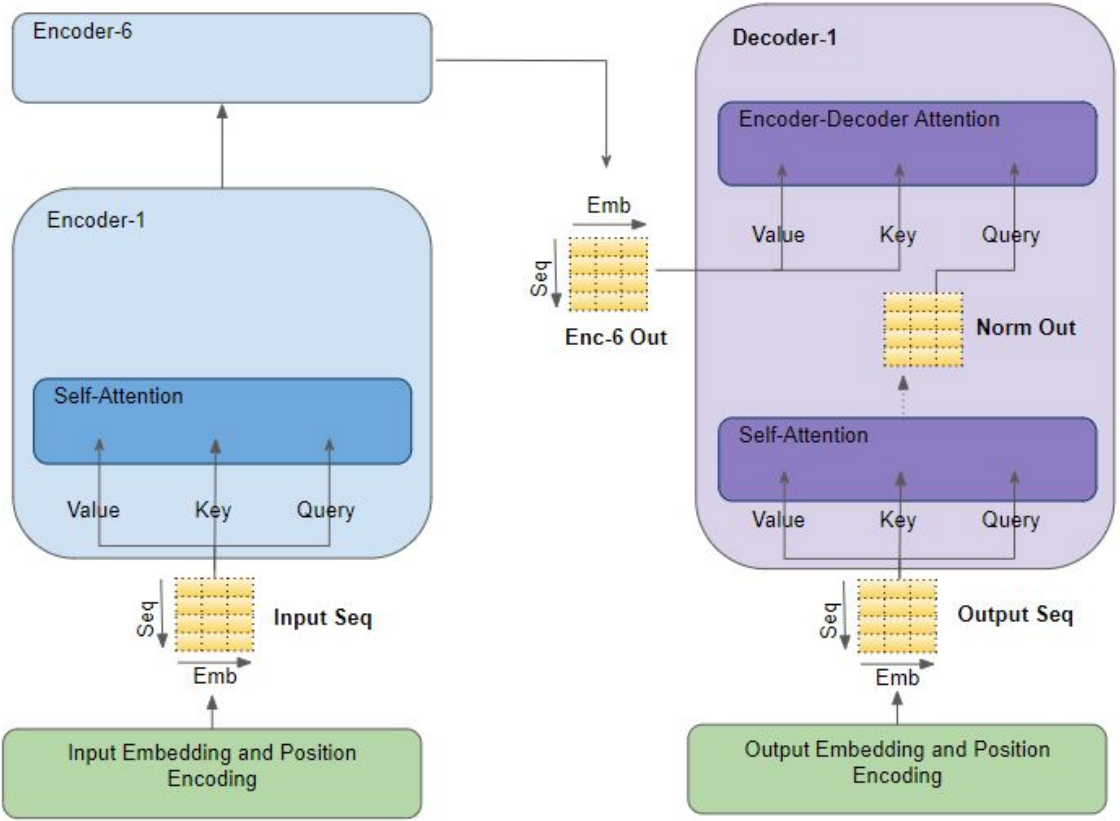
- **Query** - ερώτημα,
- **Key**- κλειδί
- **Value**- τιμή.

Και οι τρεις παράμετροι είναι παρόμοιες στη δομή, με κάθε λέξη στην ακολουθία να αντιπροσωπεύεται από ένα διάνυσμα.



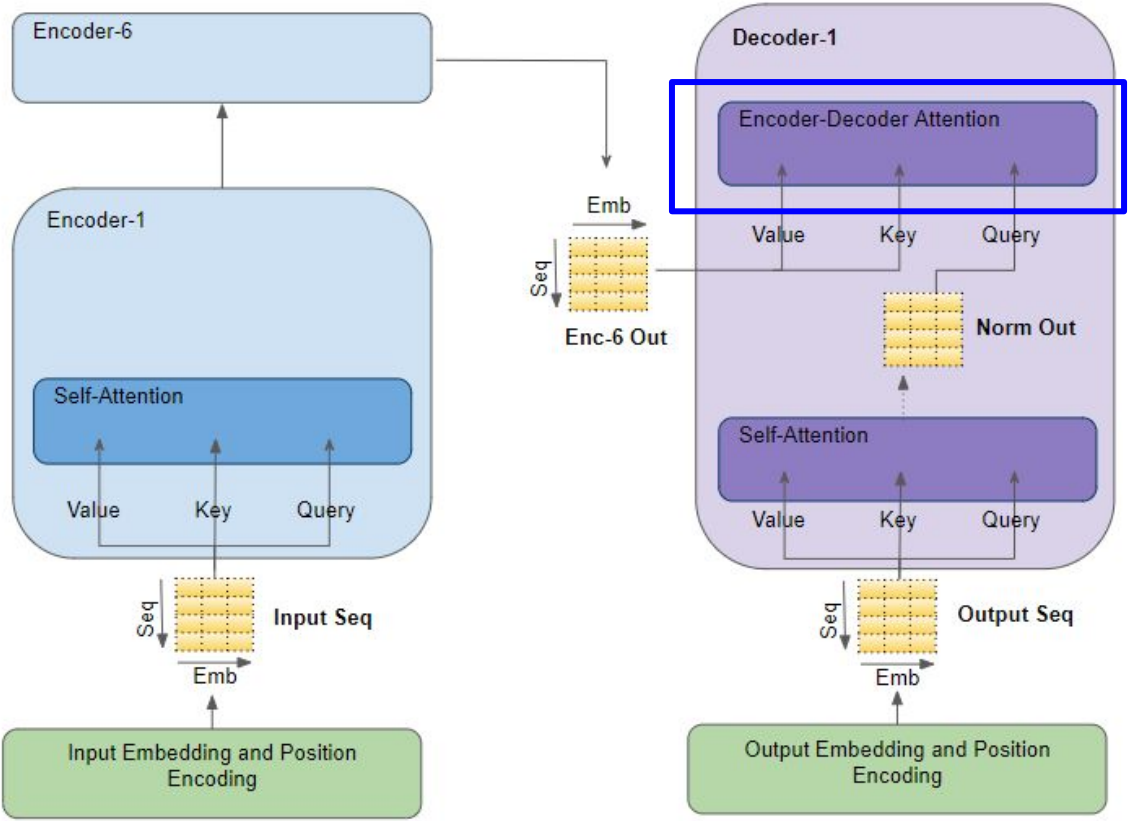
Batch of 5 Samples.
(Samples x Seq Len x Embedding size)

Encoder - Decoder: Self Attention



- Ο **Encoder 1** (ή αντίστοιχα ο **Decoder 1**) στη συνέχεια παράγει επίσης μια κωδικοποιημένη αναπαράσταση για κάθε λέξη στην ακολουθία εισαγωγής, η οποία πλέον ενσωματώνει τους βαθμούς προσοχής του encoder 1 για κάθε λέξη επίσης.
- Συνεχίσει την ίδια διαδικασία για όλους τους encoders (decoders) της στοίβας

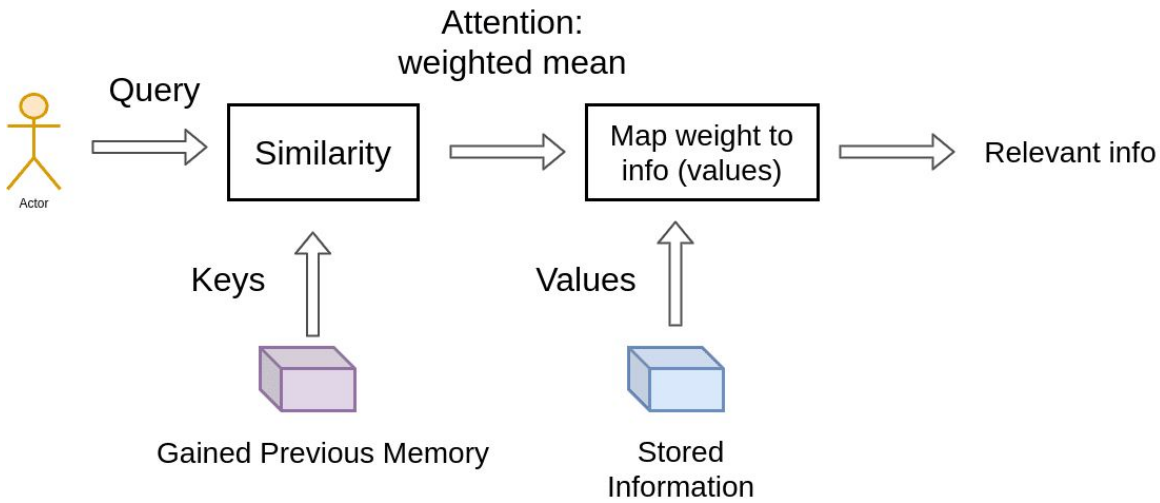
Encoder-Decoder Attention



- Ο **Encoder-Decoder Attention** λαμβάνει :
  - μια αναπαράσταση της ακολουθίας στόχου (από την Self Attention αποκωδικοποιητή)
  - μια αναπαράσταση της ακολουθίας εισόδου (από τη στοίβα Κωδικοποιητών).
- Ο Encoder-Decoder Attention παράγει μια αναπαράσταση με τις βαθμολογίες προσοχής για κάθε λέξη της ακολουθίας στόχου που καταγράφει την επιρροή των βαθμολογιών προσοχής από την ακολουθία εισόδου.

Feature-based attention

- Οι έννοιες Key, Value, και Query προέρχονται από τα συστήματα ανάκτησης πληροφοριών (information retrieval systems)
    - Ας ξεκινήσουμε με ένα παράδειγμα αναζήτησης βίντεο στο youtube.



- Όταν κάνετε αναζήτηση (query- ερώτημα) για ένα συγκεκριμένο βίντεο, η μηχανή αναζήτησης θα αντιστοιχίσει το ερώτημά σας με ένα σύνολο κλειδιών (τίτλος βίντεο, περιγραφή κ.λπ.) που σχετίζονται με πιθανά αποθηκευμένα βίντεο.
- Στη συνέχεια, ο αλγόριθμος θα σας παρουσιάσει τα καλύτερα αντιστοιχισμένα βίντεο (τιμές).

Αυτό είναι το θεμέλιο της αναζήτησης που βασίζεται σε content/features

https://theaisummer.com/transformer/

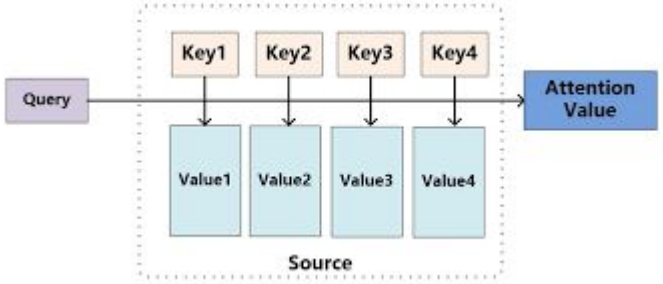
Feature-based attention

Κύρια διαφορά μεταξύ των attention συστημάτων και των συστημάτων ανάκτησης είναι ότι εισάγουμε μια πιο αφηρημένη και ομαλή έννοια της «ανάκτησης» ενός αντικειμένου.

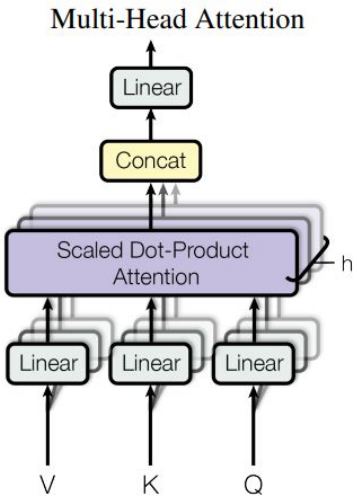
- ➔ Ορίζοντας έναν βαθμό ομοιότητας (βάρος) μεταξύ των αναπαραστάσεων μας (βίντεο για το youtube) μπορούμε να σταθμίσουμε το ερώτημά (query) μας:
  - Χωρίζουμε περαιτέρω τα δεδομένα σε ζεύγη **key-value**.
    - Χρησιμοποιούμε τα **keys** για να ορίσουμε τα **attention weights** για να δούμε τα δεδομένα
    - Χρησιμοποιούμε τα **values** ως τις πληροφορίες που θα λάβουμε πραγματικά.

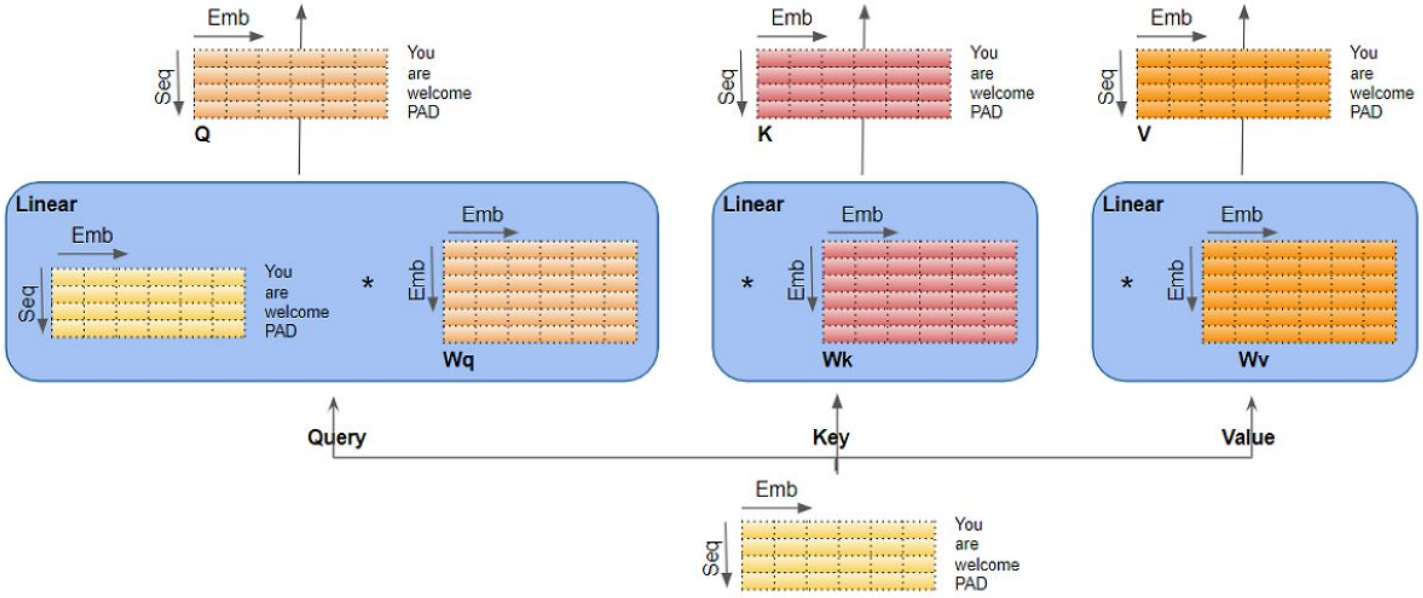
Μηχανισμός Attention

Ουσιαστικά μιμείται την ανάκτηση values που βασίζεται σε query, χρησιμοποιώντας keys σε ΒΔ με πιο fuzzy τρόπο.



Linear επίπεδα





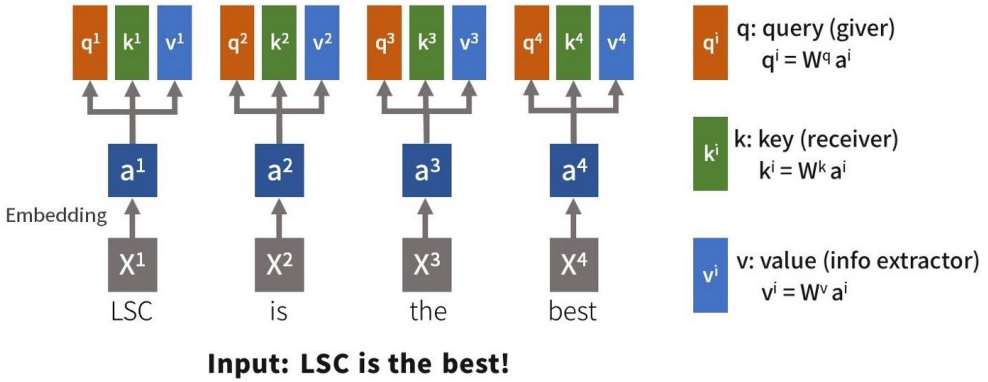
- Υπάρχουν τρία ξεχωριστά Linear επίπεδα για το Query, το Key, και το Value.
- Κάθε Linear επίπεδο έχει τα δικά του βάρη.
- Η είσοδος περνά μέσα από αυτά τα Linear επίπεδα για να παραχθούν οι πίνακες **Q, K, W**

Linear layers: αναλυτικότερα

Για κάθε είσοδο **x**, οι λέξεις στο **x** ενσωματώνονται στο διάνυσμα **a** ως είσοδο στο Self-Attention

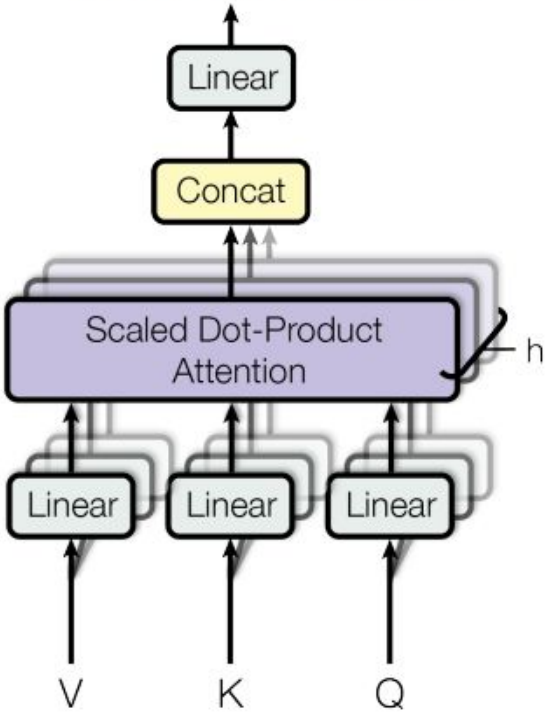
Η έξοδος των **Linear Layers** είναι οι **Query, Key** και **Value**

- q^i (Query) = $W^q a^i$
- k^i (Key) = $W^k a^i$
- v^i (Value) = $W^v a^i$



W^q, W^k, W^v είναι τα βάρη που θα καθοριστούν μέσω εκπαίδευσης

Multi-Head Attention



- ➔ Ο Transformer καλεί κάθε Attention module και το επαναλαμβάνει πολλές φορές παράλληλα.
  - ◆ Το Attention module διαχωρίζει τις παραμέτρους Query, Key, και Value N-φορές
  - ◆ Περνά κάθε τέτοιον διαχωρισμό ανεξάρτητα από μια ξεχωριστή κεφαλή
  - ◆ Συνδυάζει όλους τους υπολογισμούς attention μαζί για να παραχθεί μια τελική βαθμολογία
- ➔ Δίνει στον Transformer μεγάλη δύναμη για να κωδικοποιεί πολλαπλές σχέσεις.

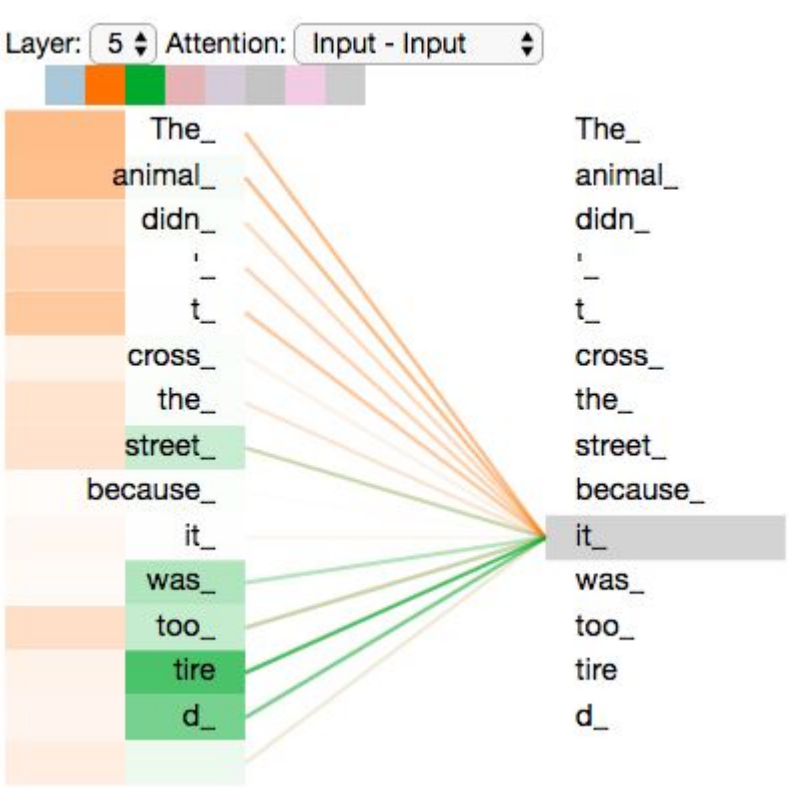
Αρχιτεκτονική Transformers: multi-head

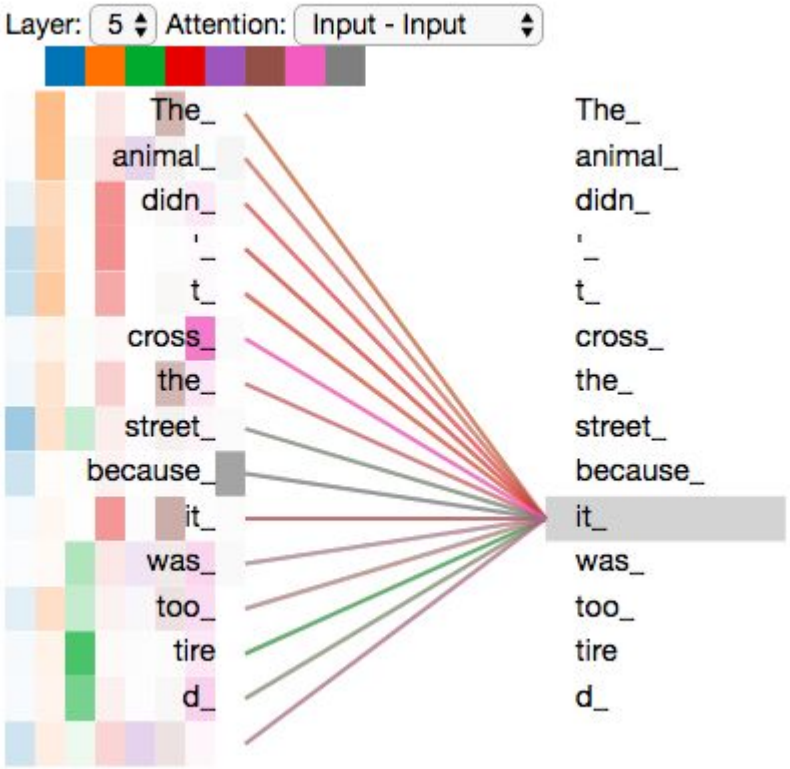
Για να μπορέσει να χειριστούν τη σημασιολογία της πρότασης, οι Transformers περιλαμβάνουν πολλαπλές βαθμολογίες προσοχής για κάθε λέξη .

π.χ.

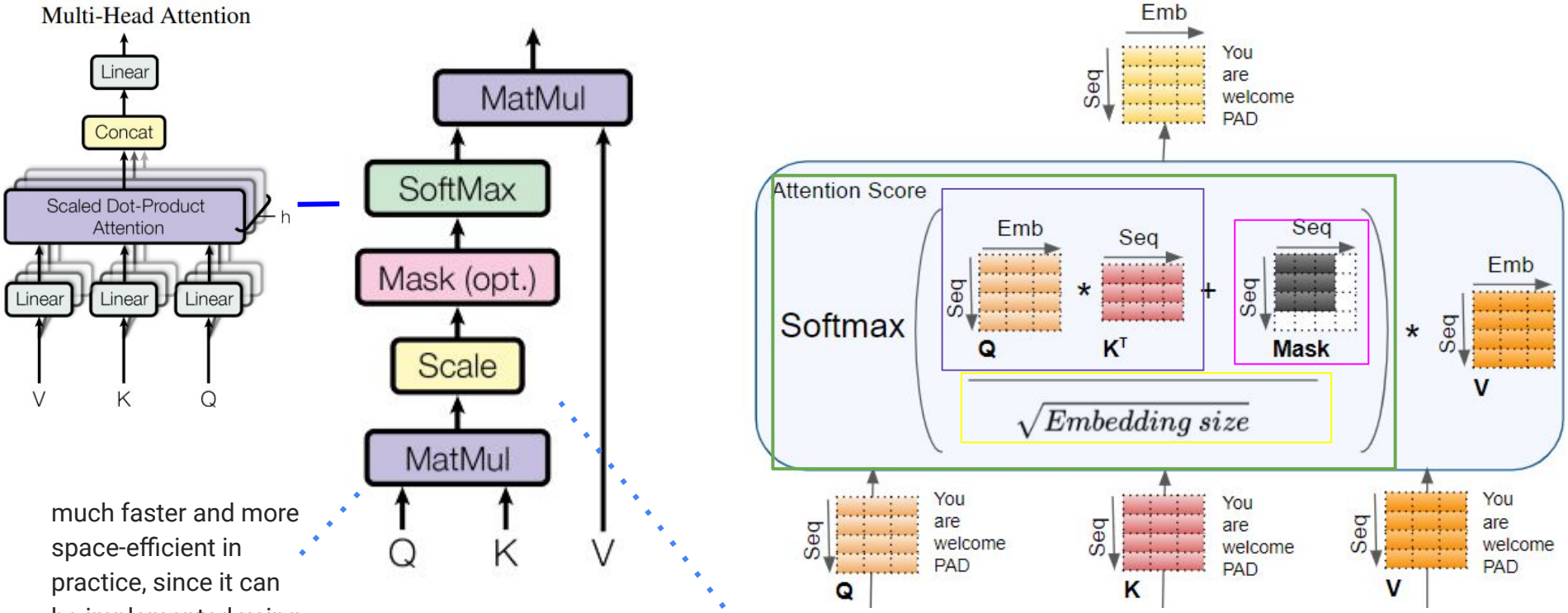
- ➔ Κατά την επεξεργασία της λέξης «it», η πρώτη βαθμολογία επισημαίνει τη λέξη «cat», ενώ η δεύτερη βαθμολογία τονίζει τη λέξη «hungry».
- ➔ Έτσι, όταν αποκωδικοποιεί τη λέξη «it», μεταφράζοντάς την σε διαφορετική γλώσσα, θα ενσωματώσει το score του «cat» και του «hungry» στη μεταφρασμένη λέξη.

Self-Attention με multi-head





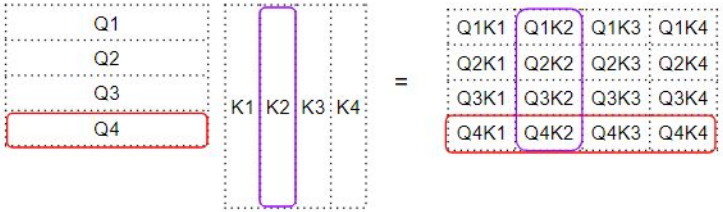
Scaled Dot-Product Attention

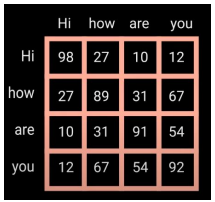


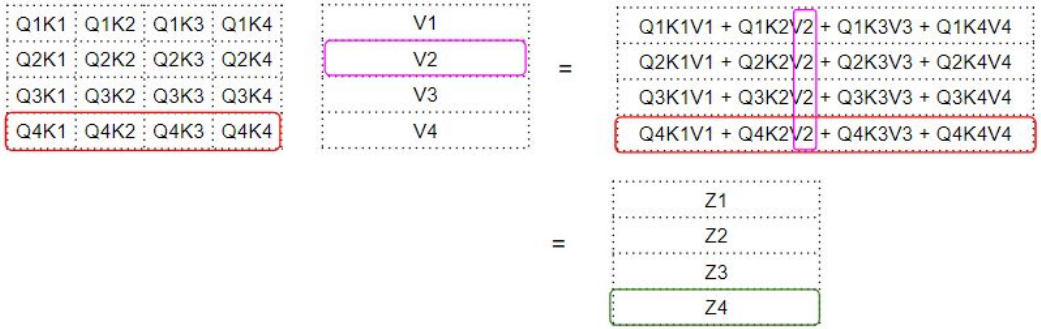
much faster and more space-efficient in practice, since it can be implemented using highly optimized matrix multiplication code

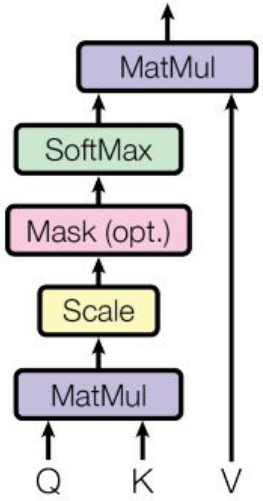
For large values of dk (embedding size), the dot products (similarity) grow large in magnitude, pushing the softmax function into regions where it has extremely small gradients. To counteract this effect, we scale the dot products by $\sqrt{dk}$

Scaled Dot-Product Attention









Fourth word Score — *Fourth Query word * first Key word*

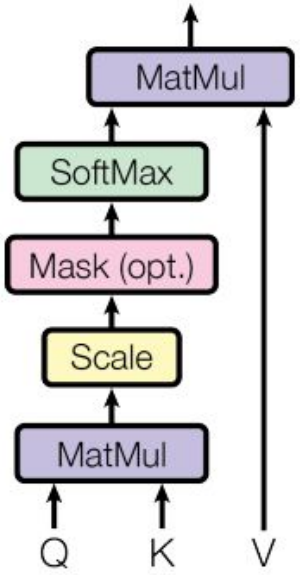
$$Z_4 = (Q_4 K_1)\, V_1 + (Q_4 K_2)\, V_2 + (Q_4 K_3)\, V_3 + (Q_4 K_4)\, V_4$$

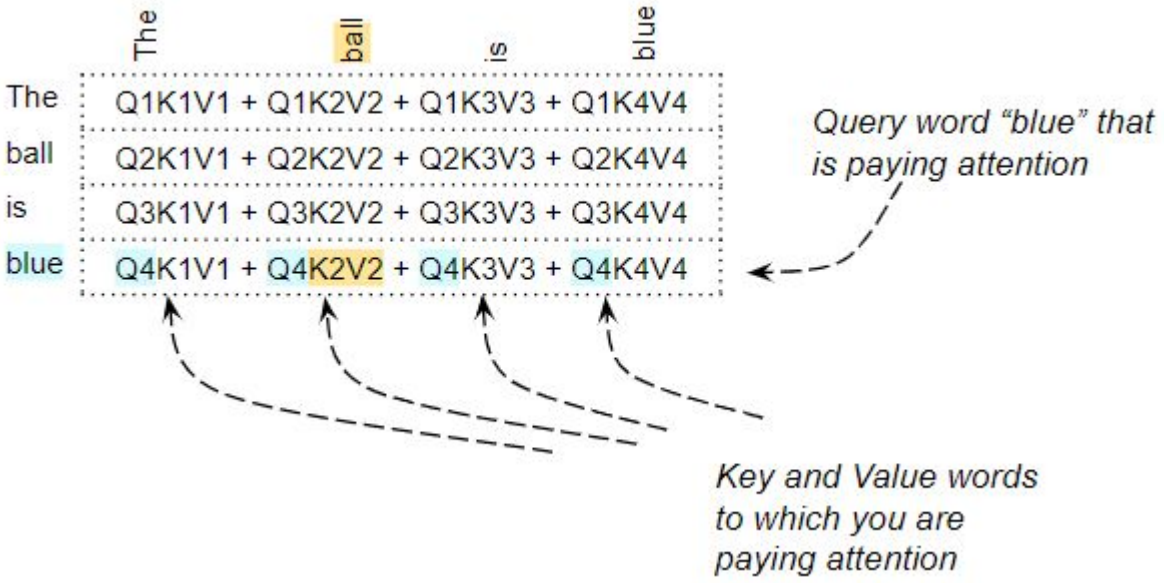
*Fourth Query word * second Key word*

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

- Το dot product μεταξύ Query και Key υπολογίζει τη συνάφεια μεταξύ κάθε ζεύγους λέξεων
- Αυτή η συνάφεια χρησιμοποιείται στη συνέχεια ως «παράγοντας» για τον υπολογισμό ενός σταθμισμένου αθροίσματος όλων των λέξεων Value
- Αυτό το σταθμισμένο άθροισμα εξάγεται ως Attention Score

Scaled Dot-Product Attention

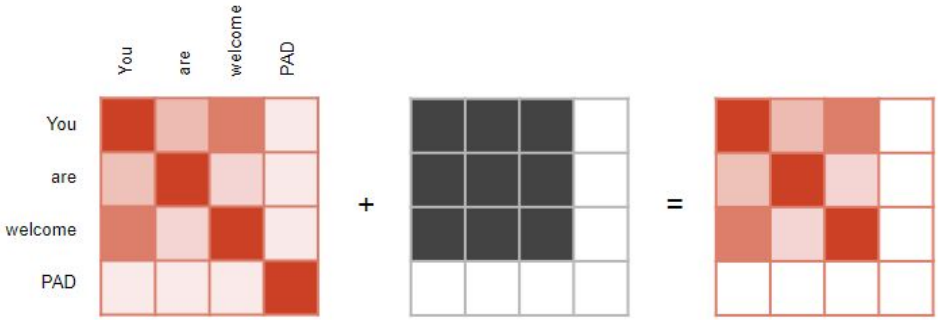




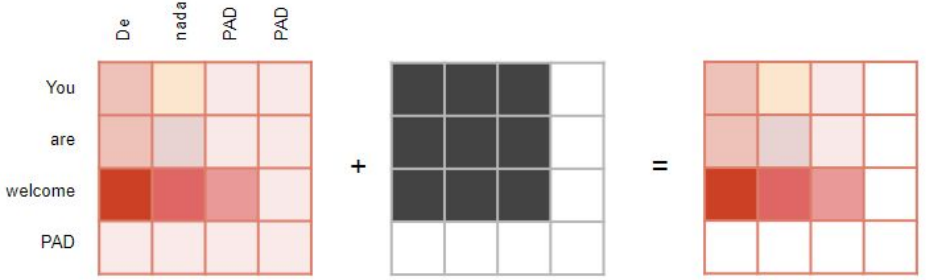
Εάν τα διανύσματα για δύο λέξεις είναι πιο ευθυγραμμισμένα, το attention score θα είναι υψηλότερο

- Θέλουμε η βαθμολογία προσοχής να είναι υψηλή για δύο λέξεις που σχετίζονται μεταξύ τους στην πρόταση.
- και θέλουμε η βαθμολογία να είναι χαμηλή για δύο λέξεις που δεν σχετίζονται μεταξύ τους.

Attention Masks



Encoder Self-Attention Scores



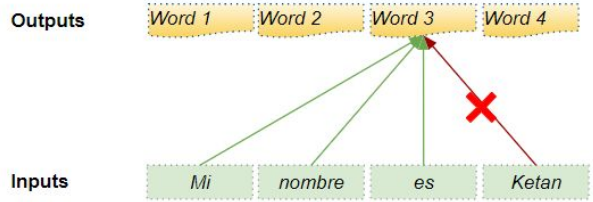
Encoder-Decoder Attention Scores

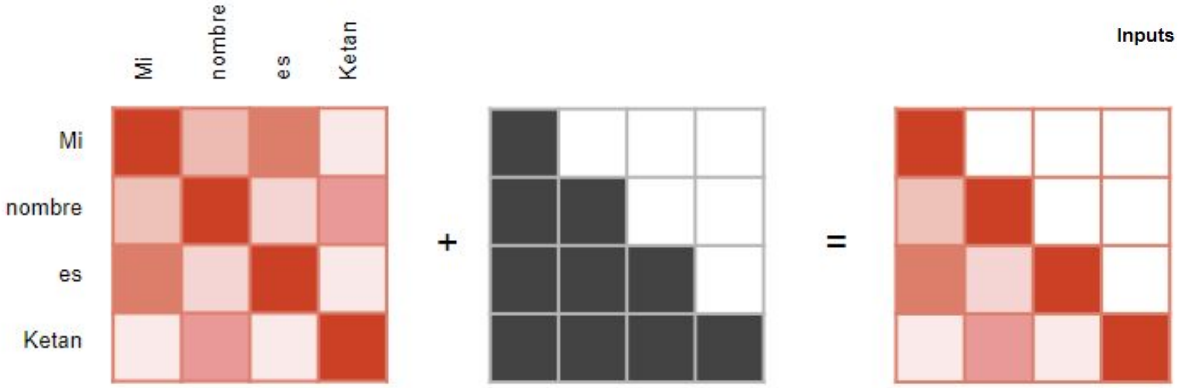
➔ Στο Encoder Self-attention και στο Encoder-Decoder-attention:

- Το masking εξυπηρετεί σε zero attention outputs όπου υπάρχει padding στις προτάσεις εισαγωγής, για να διασφαλιστεί ότι το padding δεν συμβάλλει στο self-attention.

Δεδομένου ότι οι ακολουθίες εισόδου θα μπορούσαν να έχουν διαφορετικά μήκη, επεκτείνονται με zero padding, όπως στις περισσότερες εφαρμογές NLP, ώστε ως διανύσματα σταθερού μήκους να μπορούν να εισαχθούν στον transformer.

Attention Masks



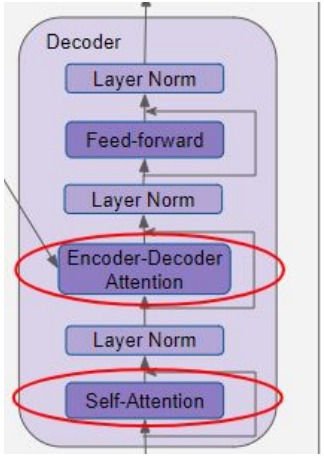


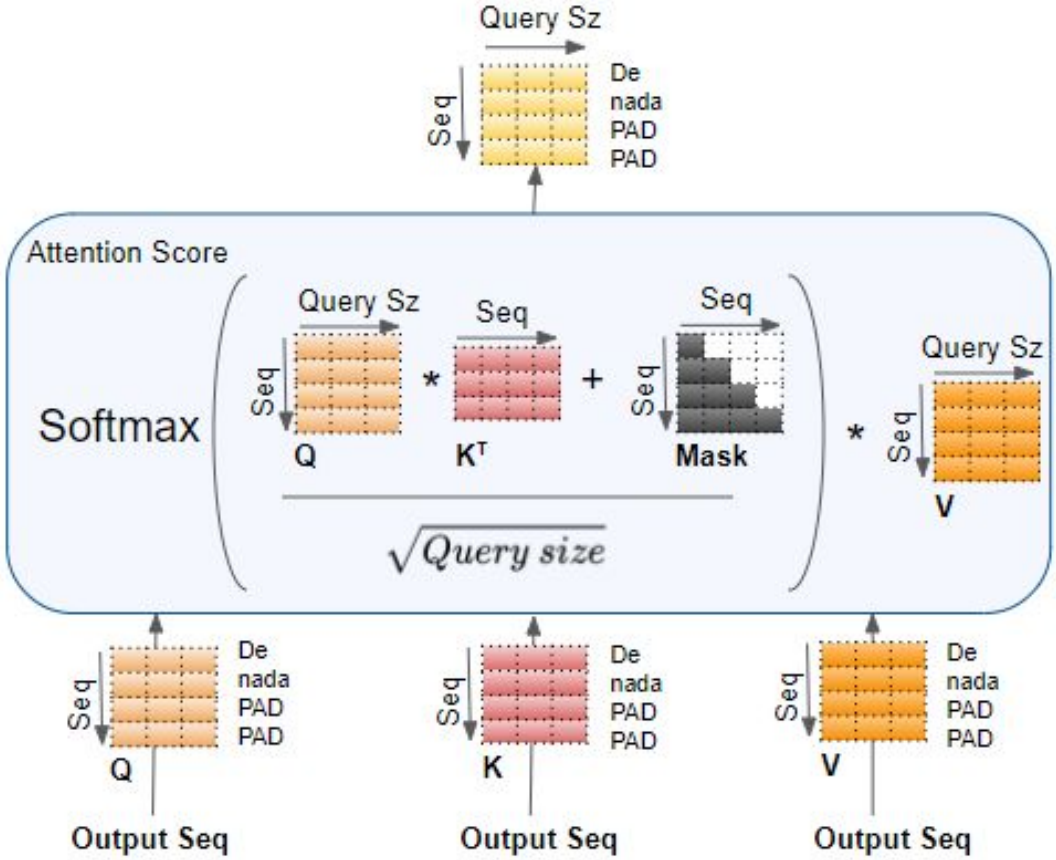
Decoder Self-Attention Scores

Στον Decoder Self-attention:

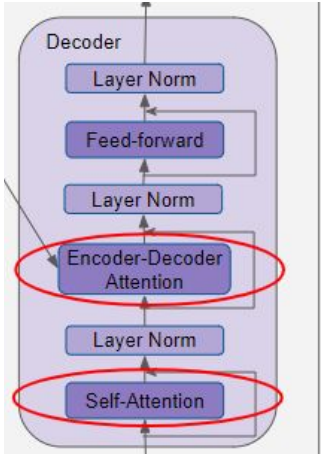
- ➔ Το masking χρησιμεύει για να αποτρέψει τον αποκωδικοποιητή από το να «κοιτάξει» μπροστά στην υπόλοιπη πρόταση-στόχο κατά την πρόβλεψη της επόμενης λέξης.
- ➔ Κατά τον υπολογισμό του score εφαρμόζεται κάλυψη στον αριθμητή ακριβώς πριν από το Softmax.
  - ◆ Τα καλυμμένα στοιχεία (λευκά τετράγωνα) ορίζονται στο αρνητικό άπειρο, έτσι ώστε το Softmax να μηδενίζει αυτές τις τιμές.

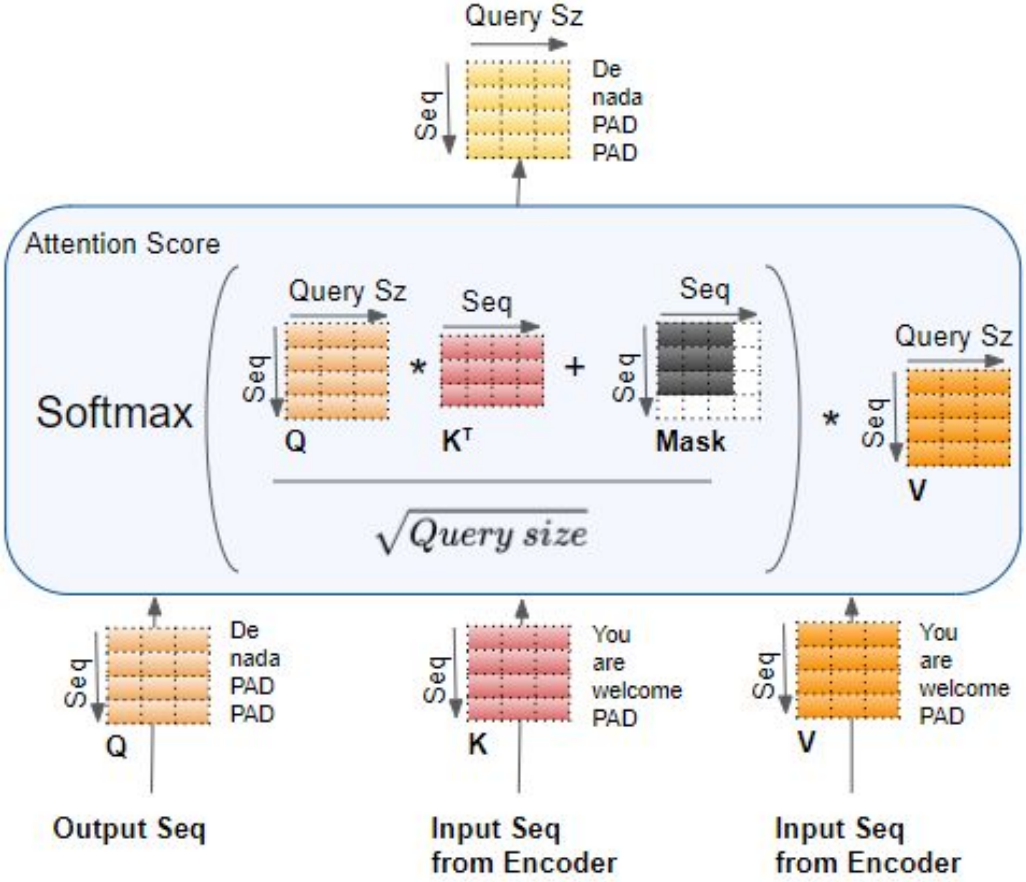
Decoder Self-Attention



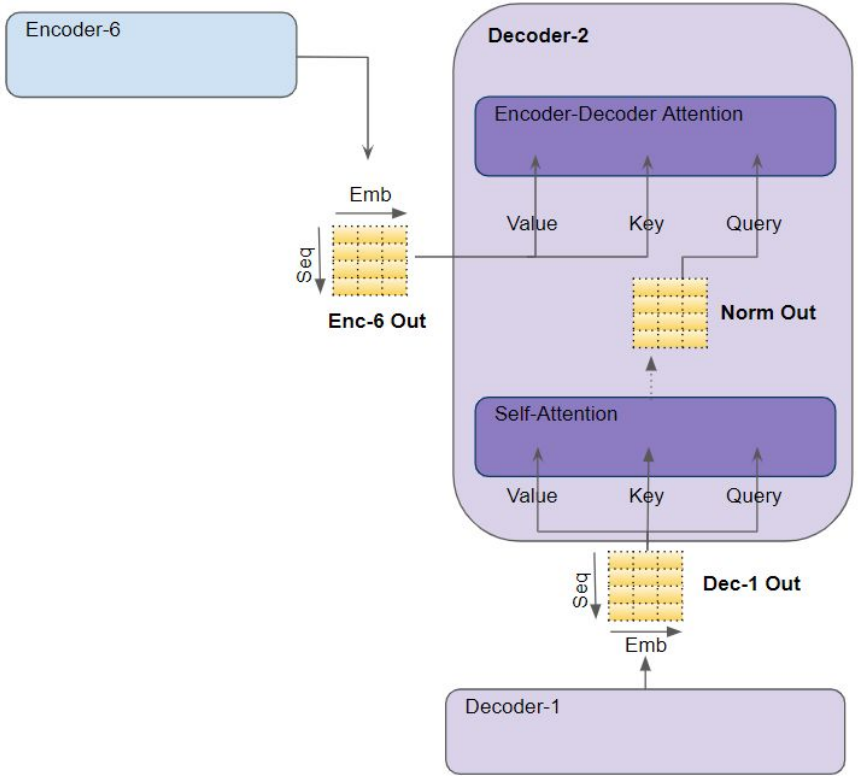


Decoder Encoder-Decoder Attention





Encoder-Decoder Attention



	La	bolal	es	azul
La	Q1K1V1 + Q1K2V2 + Q1K3V3 + Q1K4V4			
bola	Q2K1V1 + Q2K2V2 + Q2K3V3 + Q2K4V4			
es	Q3K1V1 + Q3K2V2 + Q3K3V3 + Q3K4V4			
azul	Q4K1V1 + Q4K2V2 + Q4K3V3 + Q4K4V4			

Decoder Self Attention

Target sentence paying attention to itself

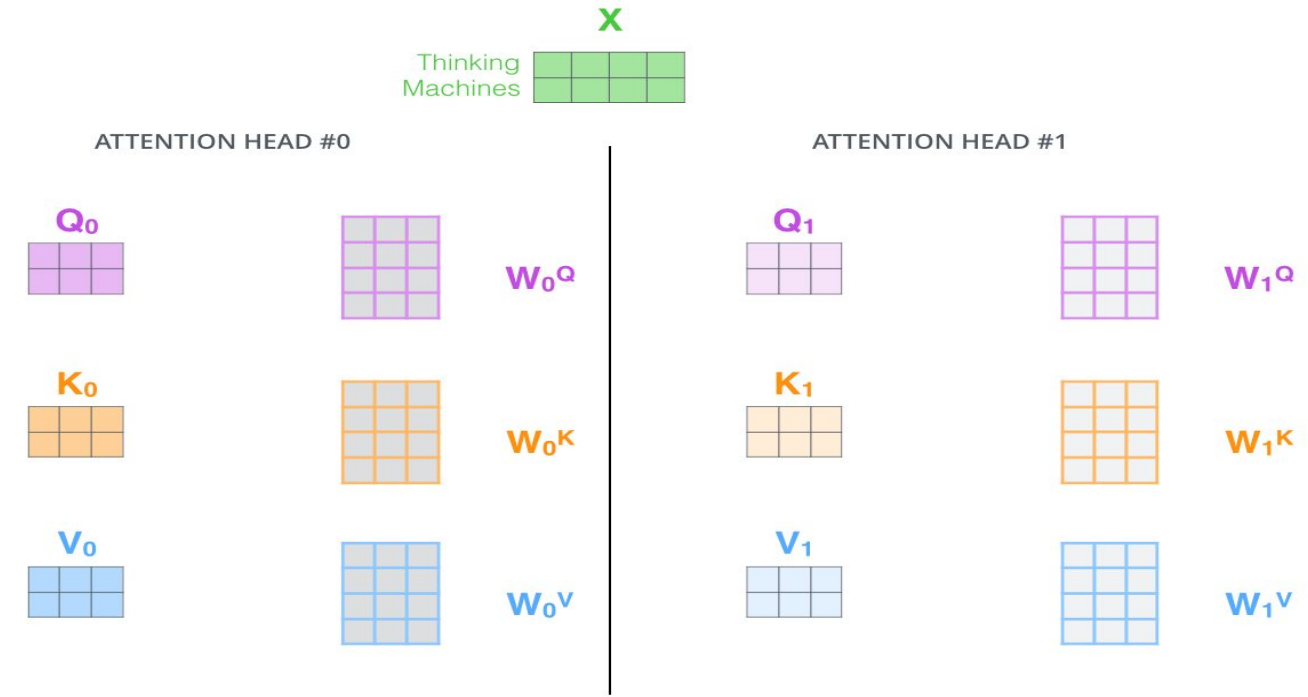
	The	ball	is	blue
La	Q1K1V1 + Q1K2V2 + Q1K3V3 + Q1K4V4			
bola	Q2K1V1 + Q2K2V2 + Q2K3V3 + Q2K4V4			
es	Q3K1V1 + Q3K2V2 + Q3K3V3 + Q3K4V4			
azul	Q4K1V1 + Q4K2V2 + Q4K3V3 + Q4K4V4			

Query word "azul" that is paying attention

Encoder-Decoder Attention

Target sentence paying attention to source sentence

Multi-head attention

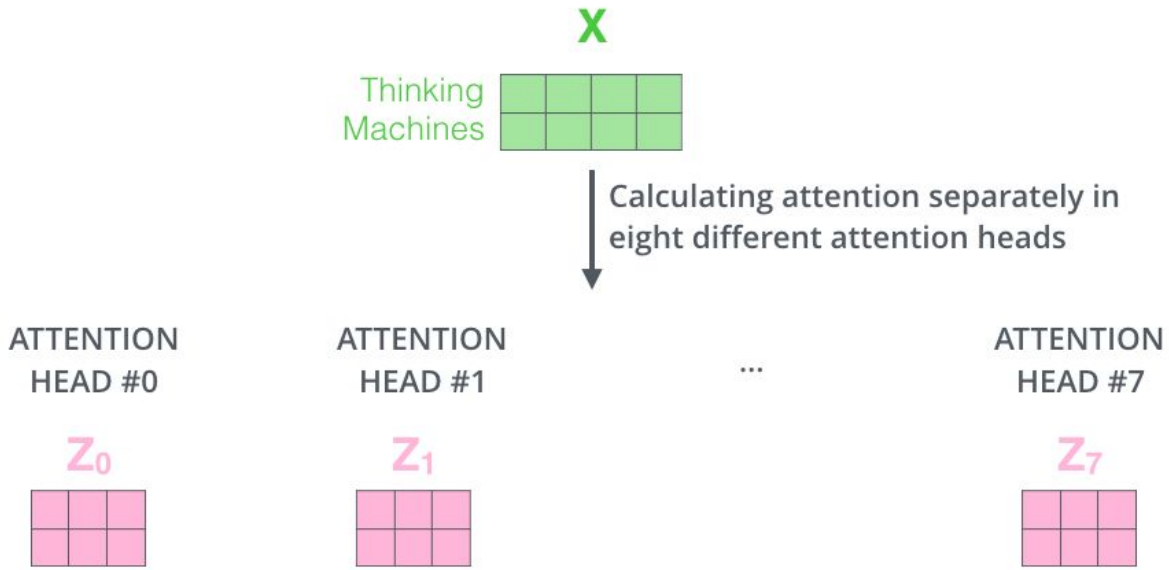


Βελτίωση του self-attention layer → προσθήκη μηχανισμού multi-head attention

➔ Διευρύνει την ικανότητα του μοντέλου να επικεντρώνεται σε διαφορετικές θέσεις.

➔ Δίνει στο attention layer «υποσυστήματα αναπαράστασης».

Multi-head attention

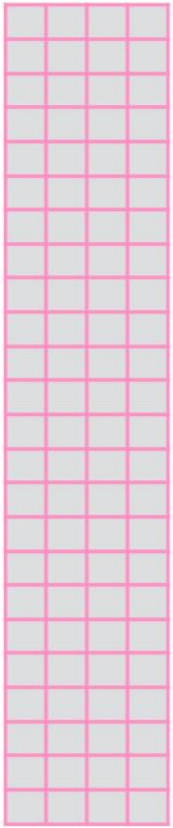


Multi-head attention

1) Concatenate all the attention heads

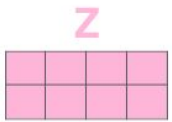
2) Multiply with a weight matrix W^O that was trained jointly with the model

X



3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN

=



1) This is our input sentence*

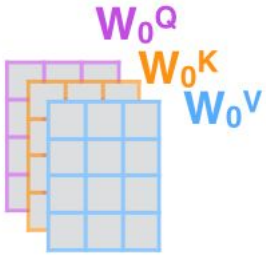
2) We embed each word*

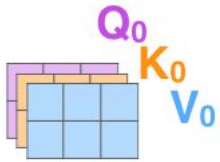
3) Split into 8 heads. We multiply X or R with weight matrices

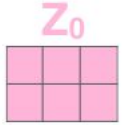
4) Calculate attention using the resulting Q/K/V matrices

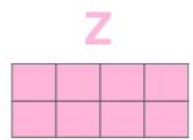
5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer

Thinking Machines

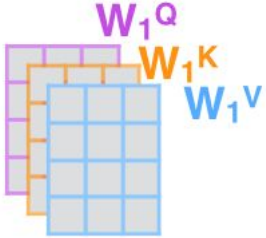


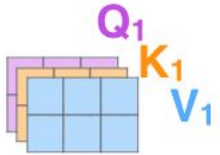


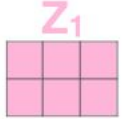


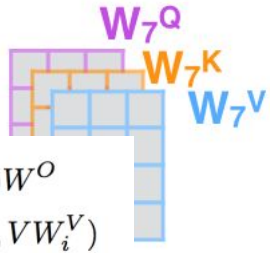


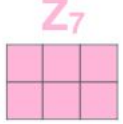
* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one











$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Σύγκριση

Το attention μειώνει τις διαδοχικές λειτουργίες και το μέγιστο μήκος διαδρομής, γεγονός που διευκολύνει τις εξαρτήσεις μεγάλης εμβέλειας

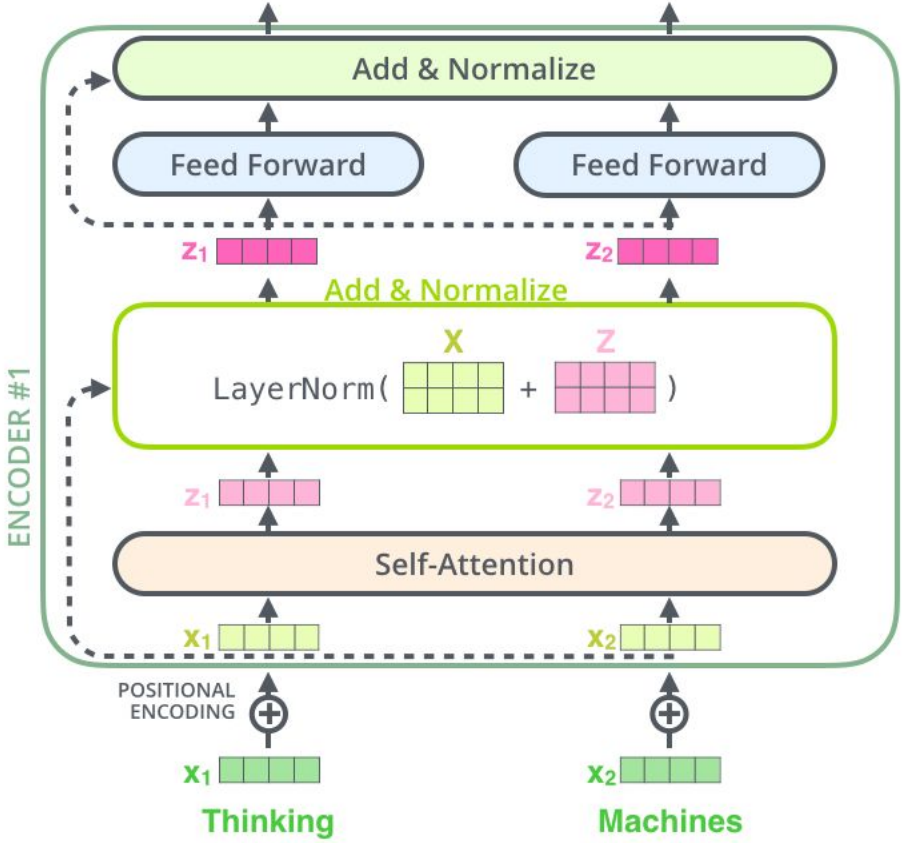
Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

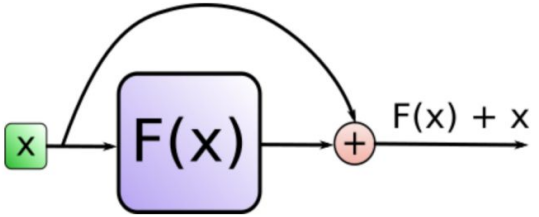
Attention Hyperparameters

- **Embedding Size** — πλάτος του embedding διανύσματος .
    - Αυτή η διάσταση μεταφέρεται σε όλο το μοντέλο του Transformer και, ως εκ τούτου, μερικές φορές αναφέρεται με άλλα ονόματα, όπως «μέγεθος μοντέλου» κ.λπ.
- **Query Size** (ίσο με μέγεθος Key και Value)
    - το μέγεθος των βαρών που χρησιμοποιούνται από τρία Linear layers για την παραγωγή των Query, Key και Value αντίστοιχα
- **Αριθμός Attention heads**
- **Batch size**, διάσταση για τον αριθμό των δειγμάτων εισόδου.

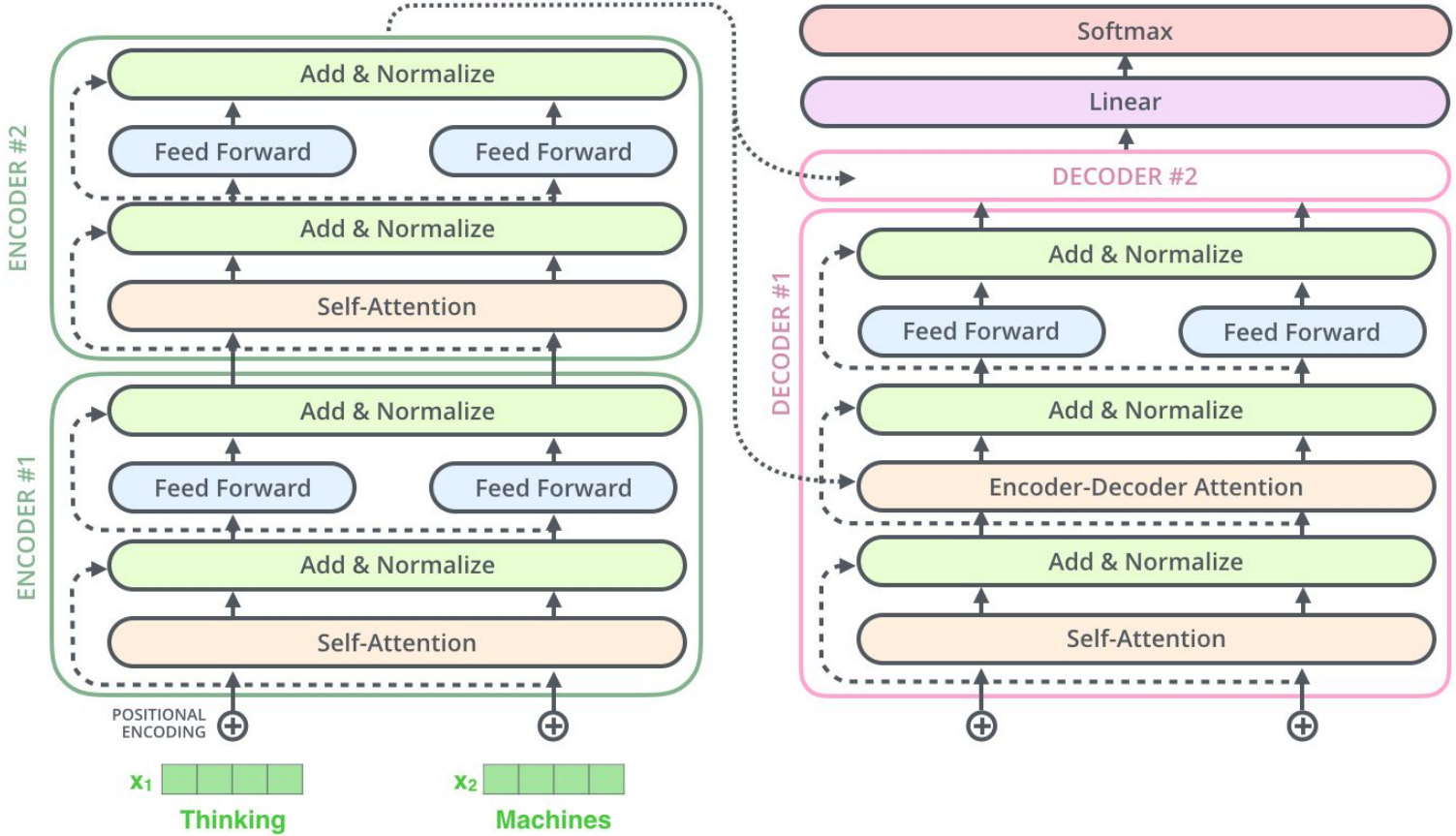
Residual Connection



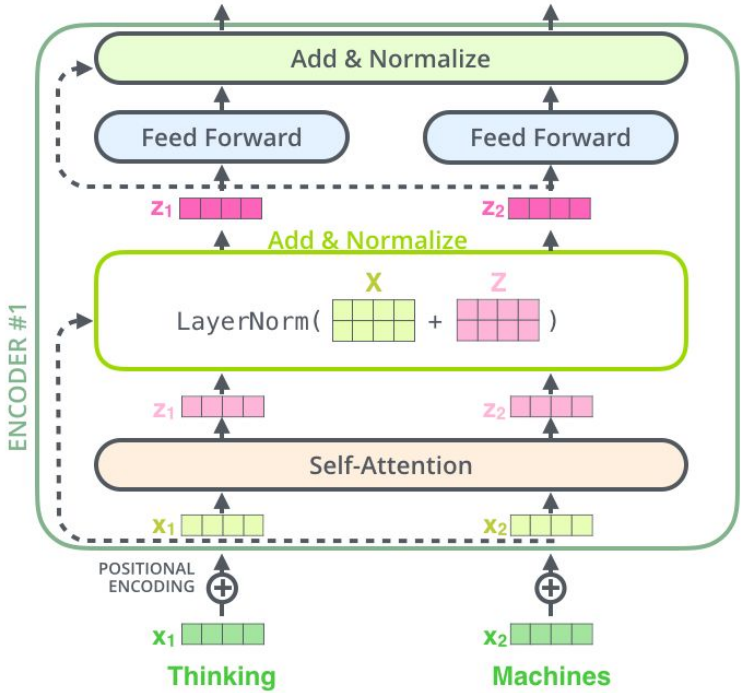
LayerNorm(x + Sublayer(x))



Residual Connection



Normalization & Position Wise Feed Forward

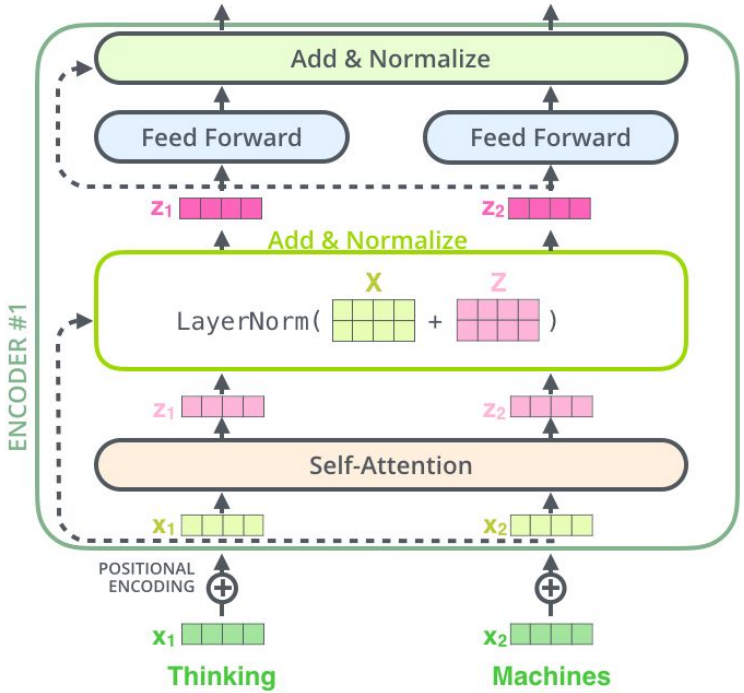


- Η κανονικοποίηση χρησιμοποιεί τα residual connections για να κοιτάξει πίσω την είσοδο του προηγούμενου επιπέδου και την έξοδο του ταυτόχρονα
  - Κανονικοποιήστε τις τιμές σε κάθε επίπεδο ώστε να έχουν 0 μέσο όρο και διακύμανση 1
  - Για κάθε κρυφή μονάδα h_i υπολογίστε $h_i \leftarrow \frac{g}{\sigma}(h_i - \mu)$

    όπου **g** είναι μια μεταβλητή, $\mu = \frac{1}{H}\sum_{i=1}^{H} h_i$ $\sigma = \sqrt{\frac{1}{H}\sum_{i=1}^{H}(h_i - \mu)^2}$

➔ Αυτό μειώνει τη “covariate shift” (δηλαδή, τα gradient dependencies μεταξύ κάθε επιπέδου) και επομένως απαιτούνται λιγότερες επαναλήψεις εκπαίδευσης

Normalization & Position Wise Feed Forward



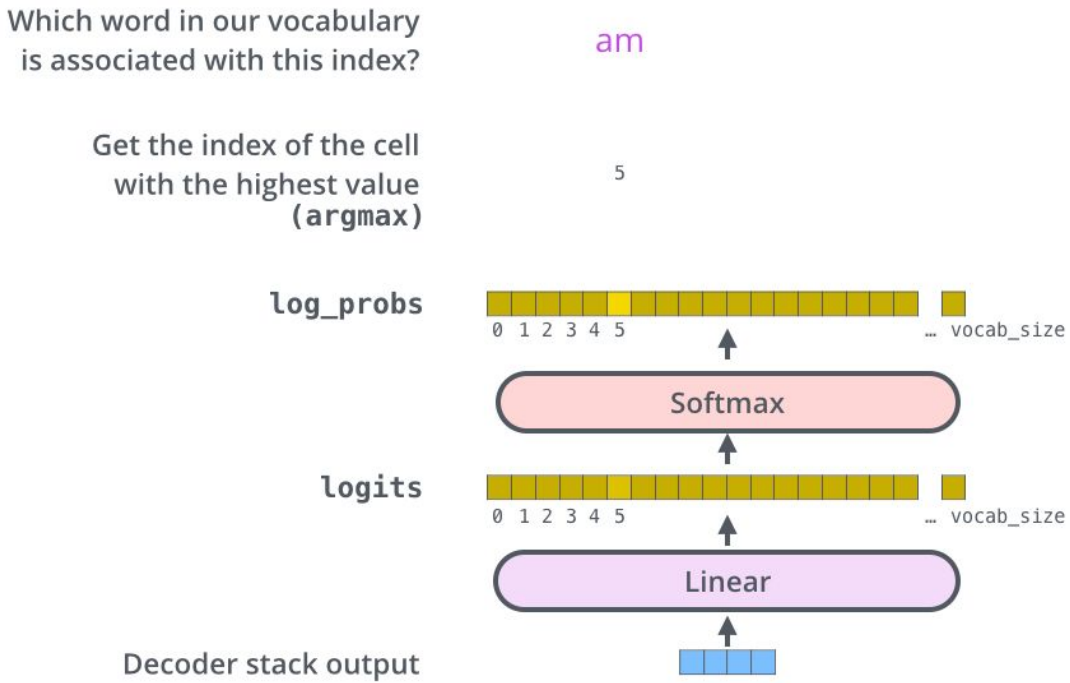
Στο μοντέλο προστίθεται ένα μικρό fully connected feed-forward network, το οποίο εφαρμόζεται σε κάθε θέση ξεχωριστά και πανομοιότυπα.

Συγκεκριμένα, το μοντέλο χρησιμοποιεί ένα Linear MLP→ReLU→Linear MLP.

Ο πλήρης μετασχηματισμός αυτού του τμήματος του residual connection μπορεί να εκφραστεί ως:

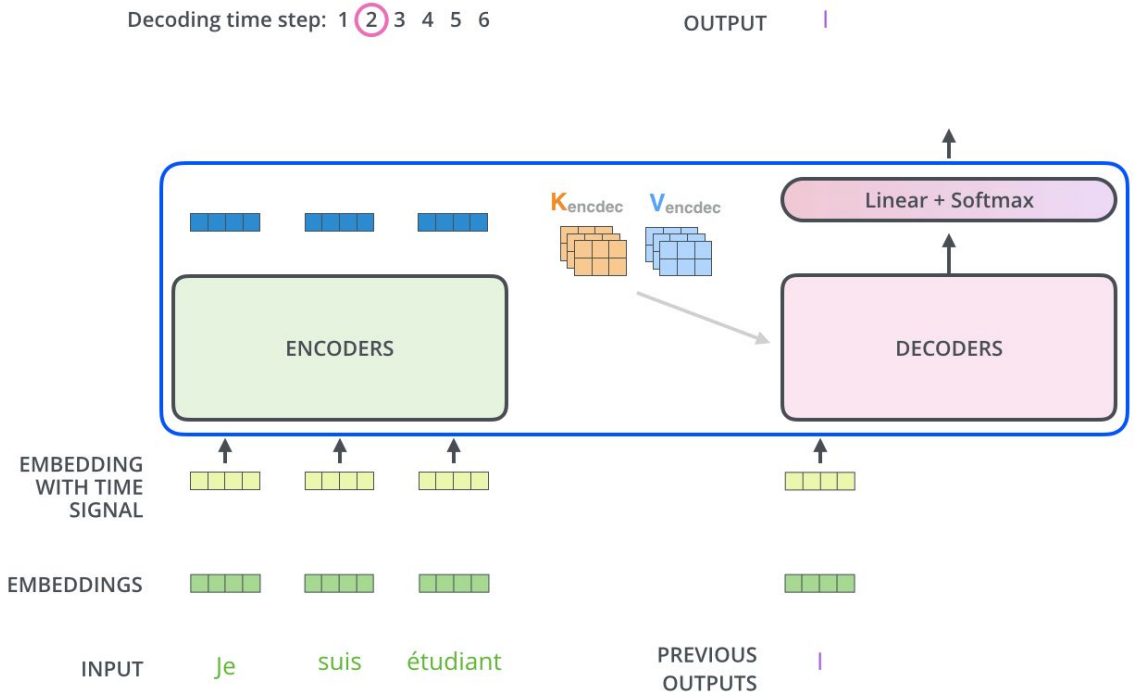
$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$
$$x = \mathrm{LayerNorm}(x + \mathrm{FFN}(x))$$

Transformer Output



- Ας υποθέσουμε ότι το μοντέλο μας γνωρίζει 10.000 μοναδικές αγγλικές λέξεις (λεξιλόγιο εξόδου) που έχει μάθει από το σύνολο δεδομένων εκπαίδευσης.
- Το decoder stack output: 10.000 κελιών (κάθε κελί αντιστοιχεί στο σκορ μιας μοναδικής λέξης).
- Στο τέλος επιλέγεται το κελί με την υψηλότερη πιθανότητα και η λέξη που σχετίζεται με αυτό παράγεται ως έξοδος για αυτό το χρονικό βήμα.

Λειτουργία transformer



- Τα attention διανύσματα K και V του τελευταίου encoder χρησιμοποιούνται από κάθε decoder στο επιπεδο encoder-decoder attention.
- Βοηθούν τον αποκωδικοποιητή να εστιάσει σε κατάλληλα σημεία στην ακολουθία εισόδου.

Παράδειγμα

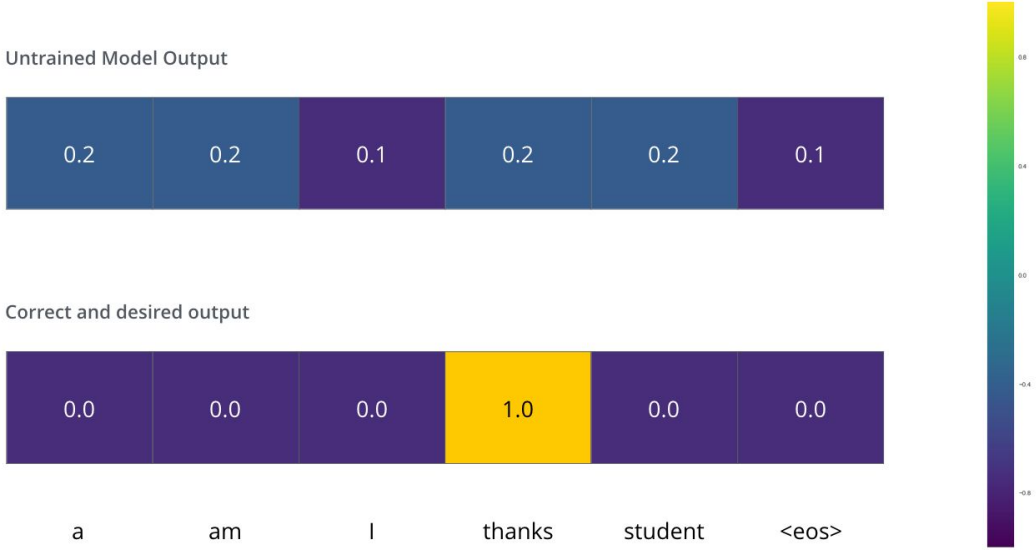
Output Vocabulary

WORD	a	am	I	thanks	student	<eos>
INDEX	0	1	2	3	4	5

One-hot encoding of the word “am”

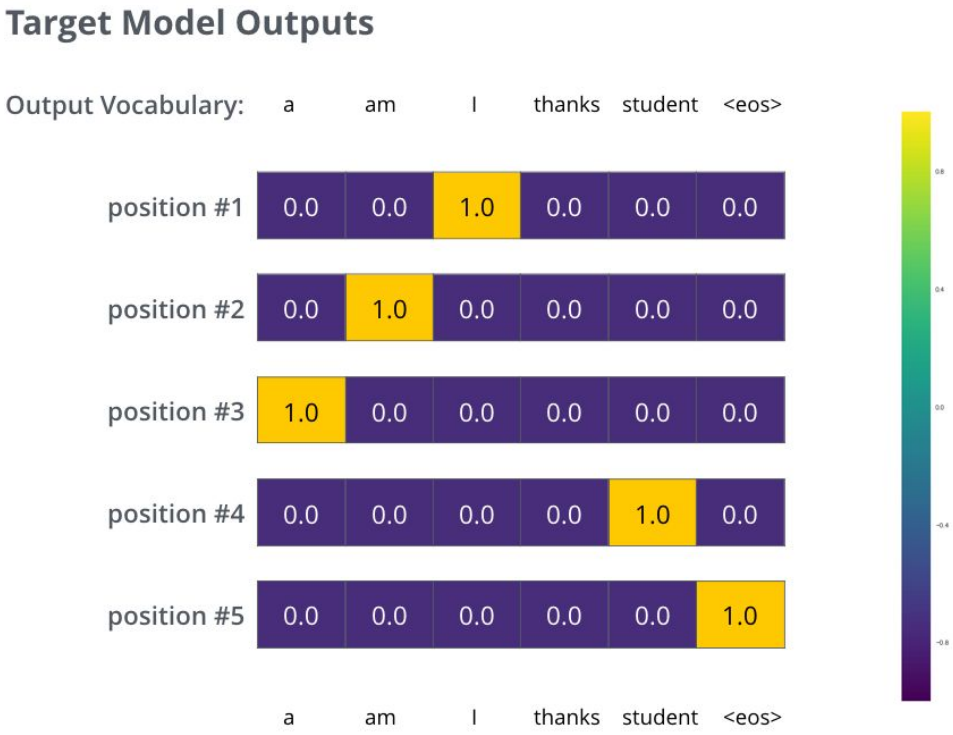
- Ας υποθέσουμε ότι το λεξικό εξόδου περιέχει 6 λέξεις (“a”, “am”, “i”, “thanks”, “student”, και “<eos>”.
- Μόλις ορίσουμε το λεξιλόγιο εξόδου μας, μπορούμε να χρησιμοποιήσουμε ένα διάνυσμα με το ίδιο πλάτος για να αντιστοιχίσουμε κάθε λέξη στο λεξιλόγιό μας.(κωδικοποίηση one-hot).

Loss Function



- Δεδομένου ότι οι παράμετροι του μοντέλου (βάρη) αρχικοποιούνται τυχαία, το (μη εκπαιδευμένο) μοντέλο παράγει μια κατανομή πιθανότητας με αυθαίρετες τιμές για κάθε κελί/λέξη.
- Μπορούμε να το συγκρίνουμε (cross entropy) με την πραγματική έξοδο και μετά να τροποποιήσουμε όλα τα βάρη του μοντέλου χρησιμοποιώντας backpropagation για να κάνουμε την έξοδο πιο κοντά στην επιθυμητή έξοδο.

Loss Function



- Θέλουμε το μοντέλο μας να εξάγει διαδοχικά τις κατανομές πιθανότητας όπου:
  - Σε κάθε κατανομή πιθανότητας αντιστοιχεί ένα διάνυσμα μεγέθου=vocab_size ((6 στο παράδειγμα μας, αλλά σε πραγματικά προβλήματα αντιστοιχεί 3.000 ή 10.000)
  - Η πρώτη κατανομή πιθανότητας έχει την υψηλότερη πιθανότητα στο κελί που συσχετίζεται με τη λέξη «i»
  - Η δεύτερη κατανομή πιθανότητας έχει την υψηλότερη πιθανότητα στο κελί που σχετίζεται με τη λέξη «am».
  - Συνεχίσουμε ώσπου η έξοδος να δείχνει το <eos>, το οποίο έχει επίσης ένα κελί που σχετίζεται με αυτό από το λεξιλόγιο των 10.000 στοιχείων.

Loss Function

- Μπορούμε να ορίσουμε τον τρόπο που το μοντέλο μας λειτουργεί
    - π.χ. μπορούμε να υποθέσουμε ότι το μοντέλο επιλέγει τη λέξη με την υψηλότερη πιθανότητα από αυτήν την κατανομή πιθανότητας και απορρίπτει τα υπόλοιπα (άπληστη αποκωδικοποίηση).

Training

Data sets:

- WMT 2014 English-German:
- 4.5 million sentences pairs with 37K tokens.
- WMT 2014 English-French:
- 36M sentences, 32K tokens.

Hardware:

- 8 Nvidia P100 GPus (Base model 12 hours, big model 3.5 days)

Results - BLEU score (bilingual evaluation understudy)

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

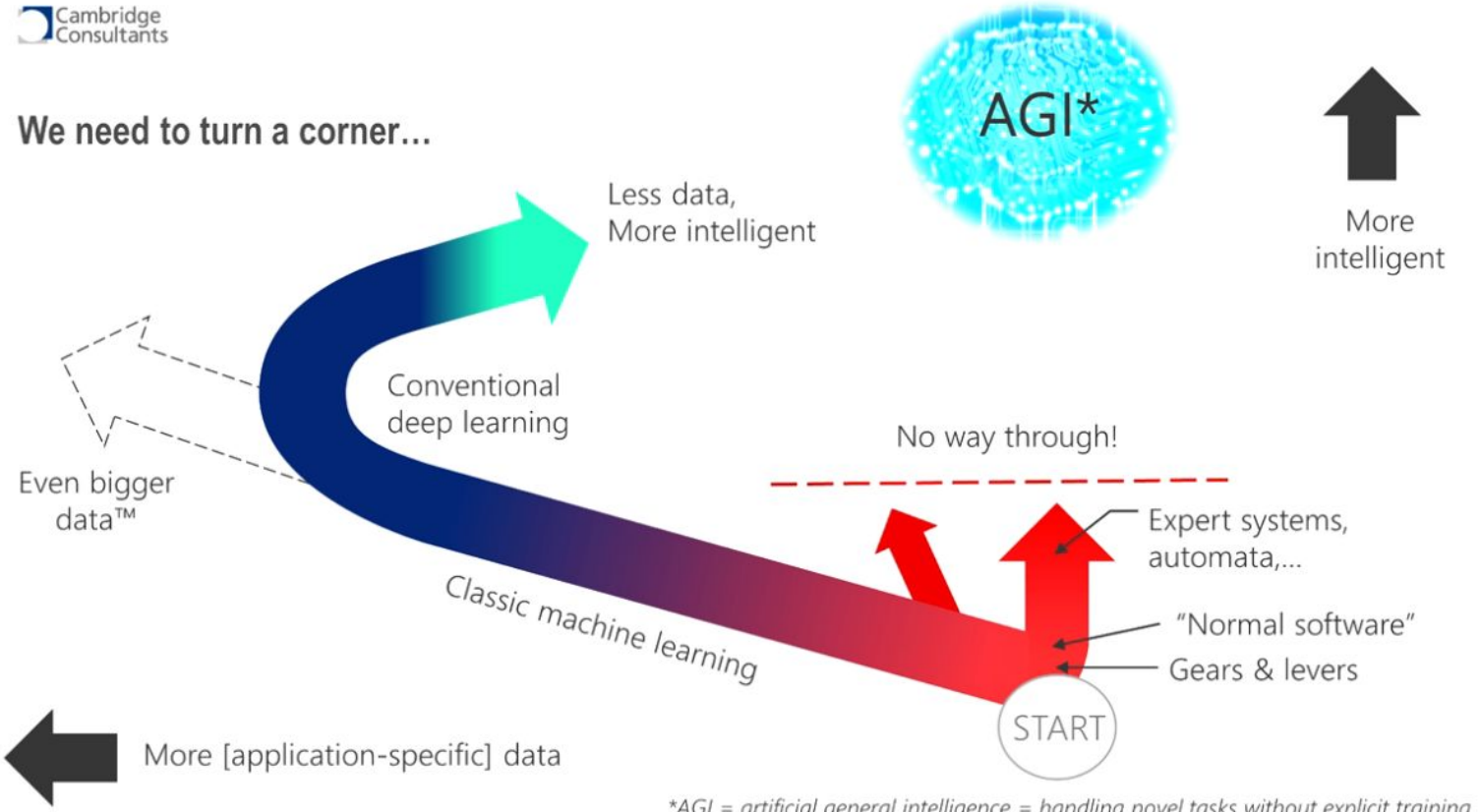
Χρήσιμοι σύνδεσμοι

Tutorial 6: Transformers and Multi-Head Attention — UvA DL Notebooks v1.1 documentation

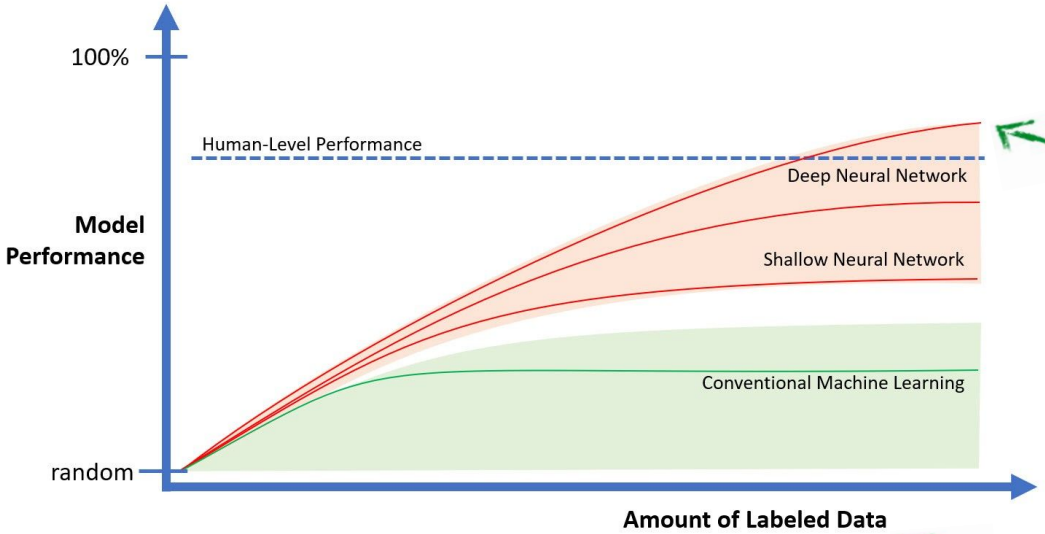
Transfer Learning

Computer Vision - Natural Language Processing

Εισαγωγή


Cambridge Consultants
We need to turn a corner…
AGI*
More intelligent
Less data,
More intelligent
Conventional deep learning
Even bigger data™
No way through!
Expert systems, automata,…
"Normal software"
Gears & levers
Classic machine learning
START
More [application-specific] data
*AGI = artificial general intelligence = handling novel tasks without explicit training

Εισαγωγή



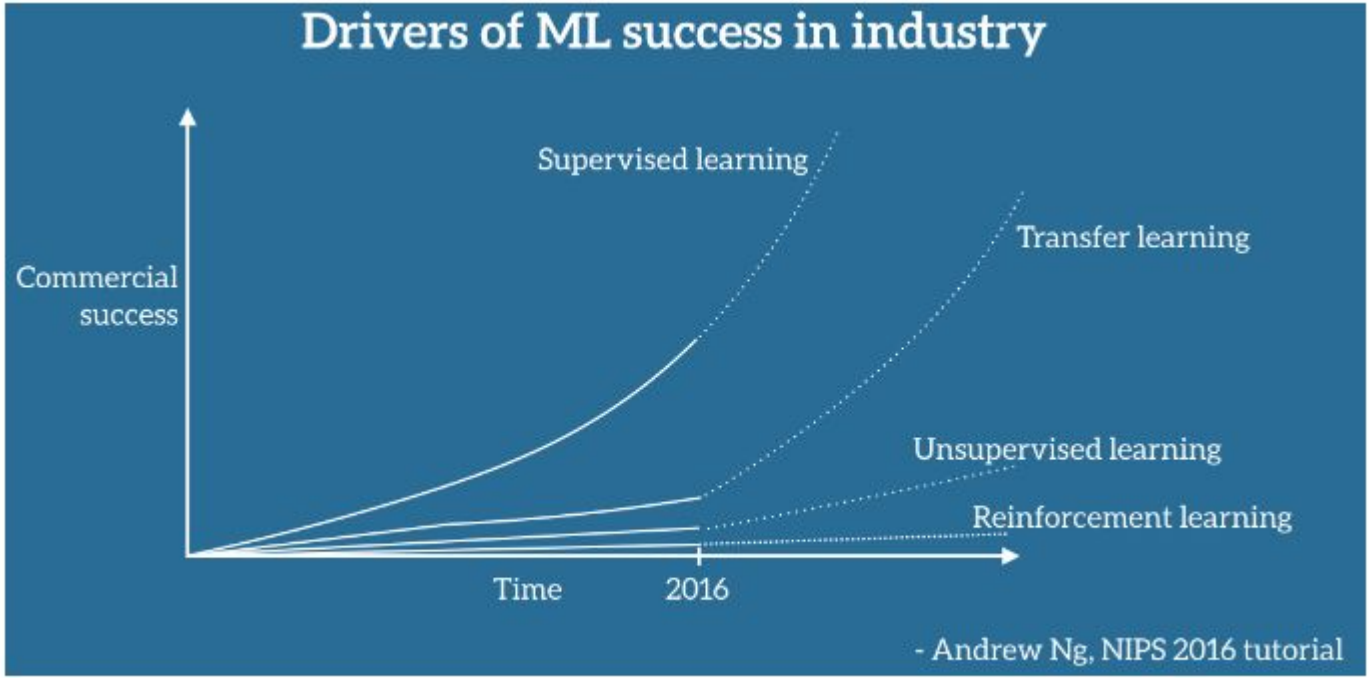
Σχεδόν γραμμική σχέση

(ο εκφραστικός χώρος του μοντέλου πρέπει να είναι αρκετά μεγάλος για να ανακαλύψει τα μοτίβα)

Δεδομένα εκπαίδευσης

Η συλλογή και επισημείωση δεδομένων μεγάλης κλίμακας και υψηλής ποιότητας είναι περίπλοκη και δαπανηρή π.χ. Imagenet

Εισαγωγή

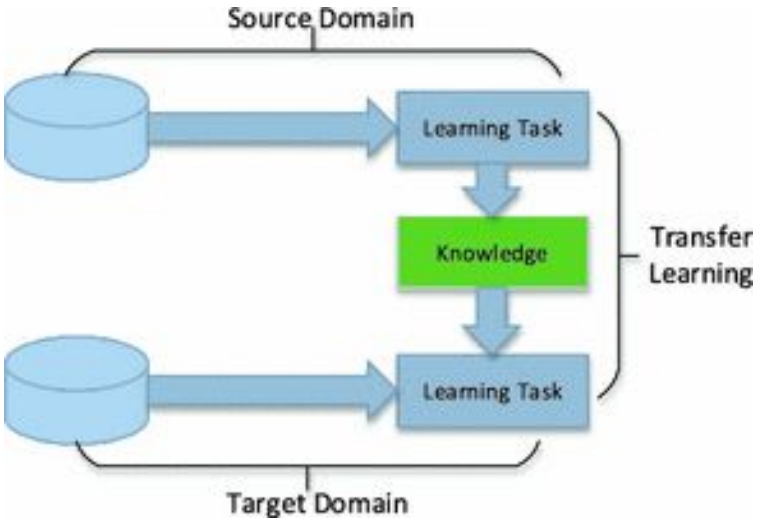


Andrew Ng 2016 NIPS

[NIPS 2016 tutorial: "Nuts and bolts of building AI applications using Deep Learning" by Andrew Ng]

"Transfer learning will be - after supervised learning - the next driver of ML commercial success" *Andrew Ng*

Βασική Ιδέα



- Επίλυση βασικού προβλήματος ανεπαρκών δεδομένων εκπαίδευσης

- Προσπαθεί να μεταφέρει τις γνώσεις από τον τομέα προέλευσης στον τομέα προορισμού χαλαρώνοντας την υπόθεση ότι τα δεδομένα εκπαίδευσης και τα δεδομένα δοκιμής πρέπει να είναι i.i.d. (independently and identically distributed)
  (τα δεδομένα εκπαίδευσης και δοκιμής έχουν διαφορετική κατανομή)

(+) Μείωση ζήτησης δεδομένων εκπαίδευσης με ετικέτες

(+) Μείωση χρόνου εκπαίδευσης

Tan C., Sun F., Kong T., Zhang W., Yang C., Liu C. (2018) A Survey on Deep Transfer Learning. Artificial Neural Networks and Machine Learning – ICANN 2018

Ορισμοί

$X=\{x_1,\ldots,x_n\} \in \mathcal{X}$: Μεταβλητές εισόδου του συστήματος (π.χ. παρατηρήσεις)

$Y=\{y_1,\ldots,y_n\} \in \mathcal{Y}$: Μεταβλητές εξόδου του συστήματος (π.χ. ετικέτες)

$\mathcal{X}, \mathcal{Y}$: Χώρος χαρακτηριστικών και ετικετών

x,y : Τιμές από το $\mathcal{X}, \mathcal{Y}$

$P(X)$: η οριακή κατανομή πιθανότητας(marginal probability distribution)

$P(Y|X)$: η υπό συνθήκη κατανομή πιθανότητας(conditional probability distribution)

Ορισμοί

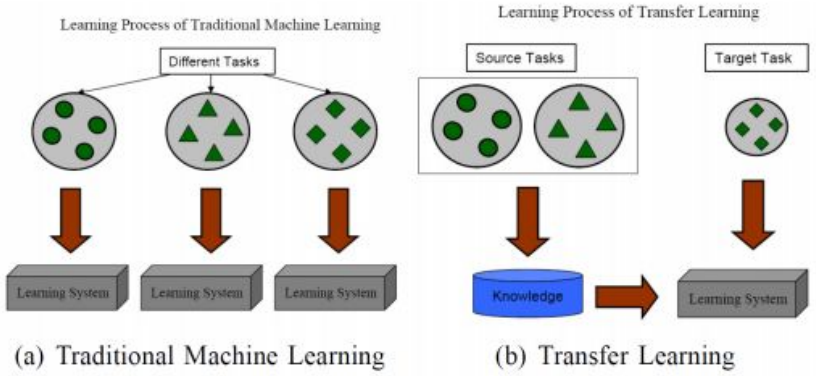
Ένας τομέας (Domain) αποτελείται από 2 στοιχεία $D=\{\mathcal{X}, P(X)\}$, όπου $\mathcal{X}$ ο χώρος χαρακτηριστικών και P(X) η οριακή κατανομή πιθανότητας(marginal probability distribution) πάνω στο χώρο των χαρακτηριστικών, με $X=\{x_1,...,x_n\} \in \mathcal{X}$.

Δεδομένου ενός τομέα, $D=\{\mathcal{X}, P(X)\}$, μια εργασία (task) T αποτελείται από έναν χώρο ετικετών $\mathcal{Y}$ και μια υπό όρους πιθανότητα κατανομής P (Y | X) που μαθαίνεται συνήθως από τα δεδομένα εκπαίδευσης $\{\{(x_i, y_i) \mid i \in \{1, 2, 3, ..., N\}$, όπου $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}\} \rightarrow T= \{\mathcal{Y}, P(Y|X)\}$

Transfer Learning: Ορισμός

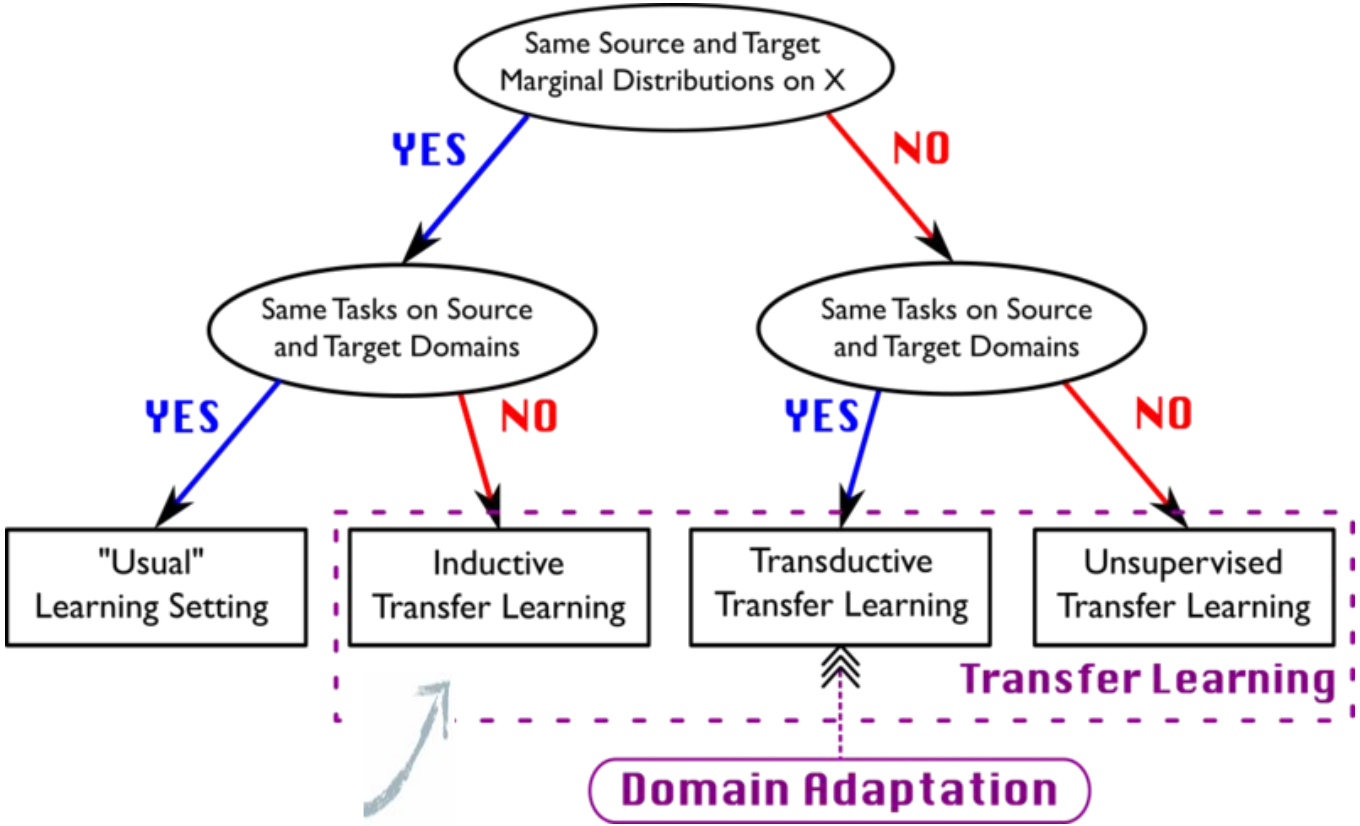
Λαμβάνοντας μια εργασία-task T_t με βάση τον τομέα προορισμού (target domain) D_t, και λαμβάνοντας τη βοήθεια από την εργασία T_s του τομέα προέλευσης (source domain) D_s, η μεταφορά μάθησης (transfer learning) στοχεύει στη μάθηση της υπό όρους κατανομής πιθανότητας $f_T=P(Y_T|X_T)$ στο D_t με τις πληροφορίες που αποκτήθηκαν από D_s και T_s, όπου $D_s \neq D_t$ ή/και $T_s \neq T_t$ και $N_s >> N_t$

Στις περισσότερες περιπτώσεις,
το μέγεθος των D_s είναι πολύ μεγαλύτερο
από το μέγεθος των D_t, $N_s \gg N_t$.



Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowl. Data Eng. 22(10), 1345–1359 (2010)

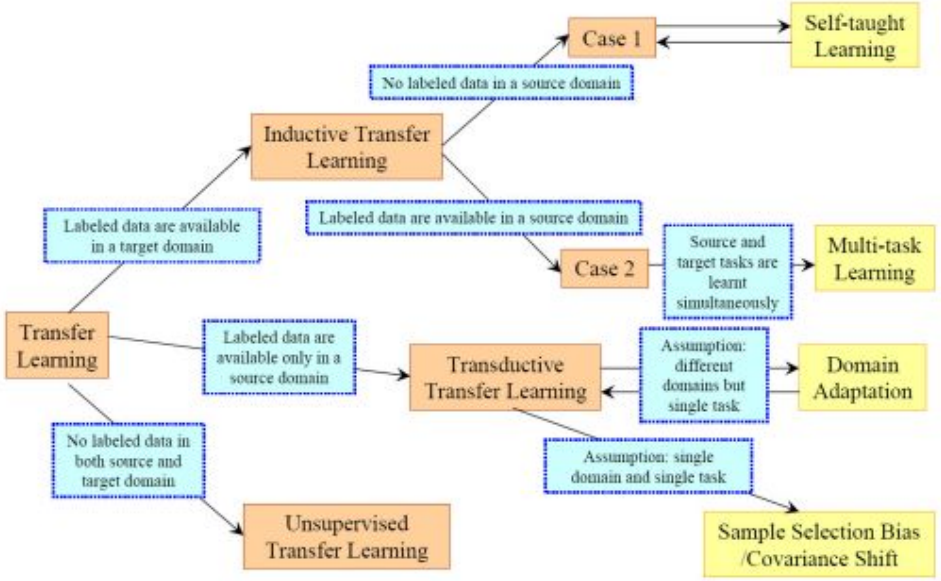
Domain Adaptation & Transfer Learning



Από το ειδικό στο γενικό

Transfer Learning: Κύριες κατηγορίες

Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowl. Data Eng. 22(10), 1345–1359 (2010)



Learning Strategy	Related Areas	Source & Target Domains	Source Domain Labels	Target Domain Labels	Source & Target Tasks	Tasks
Inductive Transfer Learning	Multi-task Learning	The Same	Available	Available	Different but Related	Regression Classification
	Self-taught Learning	The Same	Unavailable	Available	Different but Related	Regression Classification
Unsupervised Transfer Learning		Different but Related	Unavailable	Unavailable	Different but Related	Clustering Dimensionality Reduction
Transductive Transfer Learning	Domain Adaptation, Sample Selection Bias & Co-variate Shift	Different but Related	Available	Unavailable	The Same	Regression Classification

Τι να μεταφέρετε σε αυτές τις κατηγορίες;

	Inductive Transfer Learning	Transductive Transfer Learning	Unsupervised Transfer Learning
Instance-transfer	√	√	
Feature-representation-transfer	√	√	√
Parameter-transfer	√		
Relational-knowledge-transfer	√		

Instance-transfer

Δείγματα από τον τομέα προέλευσης επαναχρησιμοποιούνται μαζί με τα δεδομένα στόχου για τη βελτίωση των αποτελεσμάτων. (π.χ. κατά την επαγωγική (inductive) μεταφορά με χρήση AdaBoost)

Feature-transfer

Εντοπίζοντας καλές αναπαραστάσεις χαρακτηριστικών από τον τομέα προέλευσης που ελαχιστοποιούν το σφάλμα όταν χρησιμοποιηθούν στον τομέα προορισμού.

Parameter-transfer

Τα μοντέλα για σχετικές εργασίες μοιράζονται ορισμένες παραμέτρους ή την προηγούμενη κατανομή των υπερπαραμέτρων. Οι εργασίες προέλευσης όσο και οι εργασίες προορισμού μαθαίνονται ταυτόχρονα.

Relational-knowledge transfer

Επιχειρεί να χειριστεί δεδομένα που δεν είναι iid (π.χ. δεδομένα κοινωνικών δικτύων)

Σενάρια μεταφοράς μάθησης

$D_s \neq D_t$

- $X_s \neq X_t$

  π.χ.1 Δίγλωσσο κείμενο → cross-lingual adaptation.

  π.χ.2

- $P(X_s) \neq P(X_t)$

  π.χ. έγγραφα με διαφορετική θεματολογία → domain adaptation

$T_s \neq T_t$

- $Y_s \neq Y_t$

  π.χ. Χρήση ενός dataset 1 με κατηγορίες αντικειμένων γάτα και σκύλος για τη βελτίωση της ταξινόμησης αντικειμένων για ένα άλλο dataset με κατηγορίες καρέκλα, γραφείο και άνθρωπος.

- $P(Y_s|X_s) \neq P(Y_t|X_t)$

  π.χ. Η λέξη "monitor" σε έναν τομέα (τεχνικές αναφορές) μπορεί να χρησιμοποιείται συχνότερα ως ουσιαστικό και σε έναν άλλο τομέα (αναφορές παρακολούθησης ασθενών), μπορεί να χρησιμοποιείται κυρίως ως ρήμα.

Βασικά βήματα για μεταφορά γνώσης

Τι να μεταφέρουμε;

Πρέπει να προσδιορίσουμε:

α) ποιο τμήμα της γνώσης είναι συγκεκριμένο για τον τομέα προέλευσης και

β) τι είναι κοινό μεταξύ του τομέα προέλευσης και του τομέα προορισμού το οποίο και θα μπορούσε να μεταφερθεί.

Πότε να κάνουμε τη μεταφορά;

Πρέπει να είμαστε προσεκτικοί σχετικά με το πότε πρέπει να μεταφέρουμε και πότε όχι, με γνώμονα τη βελτίωση των αποτελεσμάτων του στόχου.

Πώς να κάνουμε τη μεταφορά;

Πρέπει να κάνουμε αλλαγές στους υπάρχοντες αλγόριθμους και να εφαρμόσουμε διαφορετικές τεχνικές.

Transfer Learning για Deep Learning

Εκπαίδευση συστήματος βαθιάς μάθησης ακόμη και αν δεν διατίθεται dataset με εκατομμύρια δεδομένα με ετικέτες.

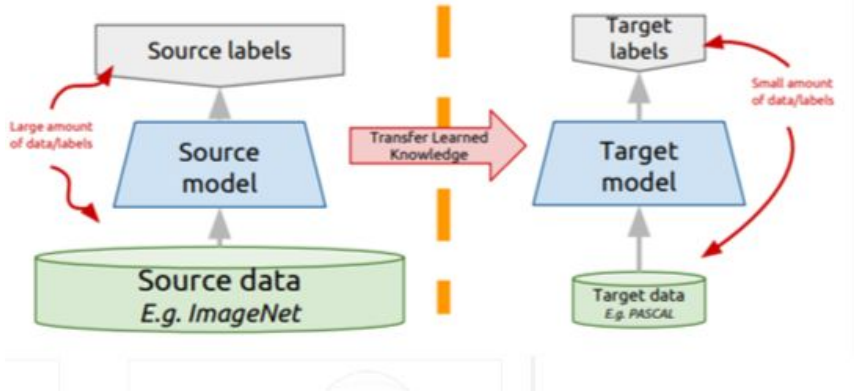
Αλλά πώς ;

- Μαθαίνοντας αναπαραστάσεις από δεδομένα χωρίς ετικέτες.
- Εκπαιδεύοντας κοντινούς τομείς με τον τομέα προορισμού, από τους οποίους είναι εύκολο να δημιουργηθούν ετικέτες.
- Μεταφέροντας αναπαράσταση της γνώση από σχετικές εργασίες.

Transfer Learning για Deep Learning

Ορισμός

Με δεδομένη μια εργασία Transfer Learning που ορίζεται από $\langle D_s, T_s, D_t, T_t, f_T(\cdot)\rangle$, η μεταφορά μάθησης (transfer learning) στοχεύει στη μάθηση της μη γραμμικής συνάρτησης f_T που αντικατοπτρίζει ένα βαθύ νευρωνικό δίκτυο.



Tan C., Sun F., Kong T., Zhang W., Yang C., Liu C. (2018) A Survey on Deep Transfer Learning. Artificial Neural Networks and Machine Learning – ICANN 2018

Transfer Learning για Deep Learning: Κατηγορίες

Instances-based deep transfer learning

Χρήση δειγμάτων του τομέα προέλευσης με κατάλληλο βάρος

Mapping-based deep transfer learning

Χαρτογράφηση δειγμάτων από δύο τομείς σε έναν νέο χώρο δεδομένων με καλύτερη ομοιότητα

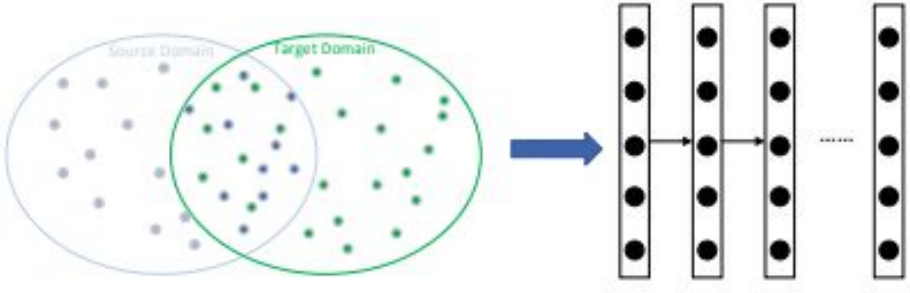
Network-based deep transfer learning

Επαναχρησιμοποίηση του τμήματος του δικτύου που έχει ήδη εκπαιδευτεί στον τομέα προέλευσης

Adversarial-based deep transfer learning

Χρησιμοποίηση της τεχνολογίας αντιπαράθεσης (adversarial) για να βρεθούν μεταβιβάσιμα χαρακτηριστικά που να είναι κατάλληλα και για τους δύο τομείς.

Instances-based deep transfer learning

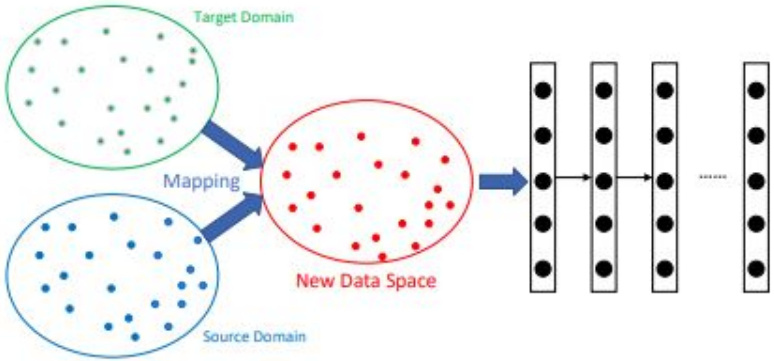


Αναφέρεται στη χρήση μιας συγκεκριμένης στρατηγικής προσαρμογής βάρους:

➔ Επιλέγονται μερικά δείγματα από τον τομέα προέλευσης ως συμπληρώματα στην εκπαίδευση που έχει οριστεί στον τομέα προορισμού , εκχωρώντας κατάλληλες τιμές βάρους σε αυτά τα επιλεγμένα δείγματα.

Βασίζεται στην υπόθεση ότι «Παρόλο που υπάρχουν διαφορές μεταξύ των δύο τομέων, μερικές εμφανίσεις στον τομέα προέλευσης μπορούν να χρησιμοποιηθούν από τον τομέα προορισμού με τα κατάλληλα βάρη».

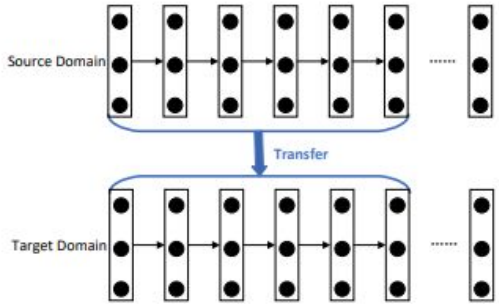
Mapping-based deep transfer learning



➔ Αναπαράσταση δειγμάτων από τον τομέα προέλευσης και τον τομέα προορισμού σε ένα νέο χώρο δεδομένων και είσοδο σε ένα κοινό δίκτυο

Βασίζεται στην υπόθεση ότι «Παρόλο που υπάρχουν διαφορές μεταξύ των τομέων προέλευσης και προορισμού, τα δείγματα μπορούν να είναι παρόμοια σε έναν περίπλοκο νέο χώρο δεδομένων».

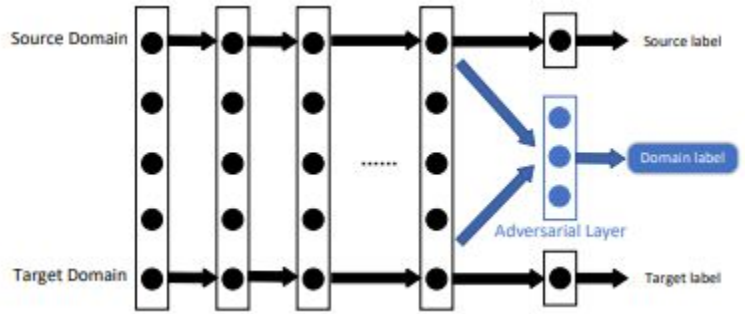
Network-based deep transfer learning



➔ Επαναχρησιμοποίηση μέρος του δικτύου που προ-εκπαιδεύτηκε στον τομέα προέλευσης, συμπεριλαμβανομένης της δομής του δικτύου και των παραμέτρων σύνδεσης και μεταφοράς αυτού ώστε να αποτελέσει μέρος ενός δικτύου βαθιάς μάθησης που χρησιμοποιείται στον τομέα προορισμού.

Βασίζεται στην υπόθεση ότι «Το νευρωνικό δίκτυο είναι παρόμοιο με το μηχανισμό επεξεργασίας του ανθρώπινου εγκεφάλου, ως μια επαναληπτική και συνεχής αφαιρετική διαδικασία. Τα μπροστινά στρώματα του δικτύου μπορούν να θεωρηθούν ως εξαγωγείς αφαιρετικών χαρακτηριστικών"

Adversarial-based deep transfer learning



➔ Εμπνευσμένο από τα Γεννητικά Ανταγωνιστικά Δίκτυα (GAN) βρίσκει μεταφέρσιμες αναπαραστάσεις που μπορούν να εφαρμοστούν τόσο στον τομέα προέλευσης όσο και στον τομέα προορισμού.

Βασίζεται στην υπόθεση ότι «Για αποτελεσματική μεταφορά, η καλή αναπαράσταση πρέπει να είναι διακριτική για το κύριο στόχο εκπαίδευσης και να διακρίνεται μεταξύ του τομέα προέλευσης και του τομέα στόχου».

Στρατηγικές Deep Transfer Learning

➔ Προεκπαιδευμένα μοντέλα για εξαγωγή χαρακτηριστικών

Off-the-shelf Pre-trained Models as fixed Feature Extractors

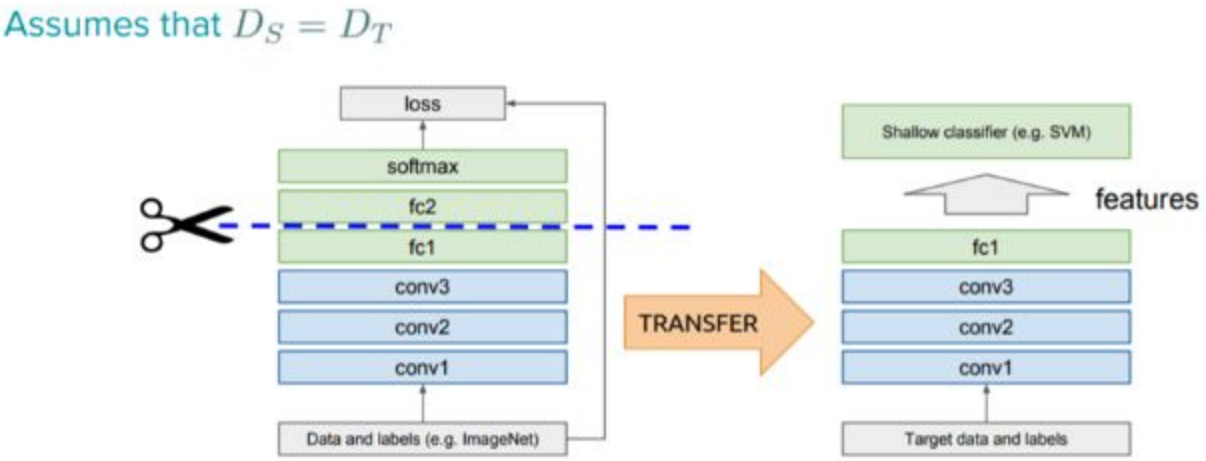
➔ Ακριβής προσαρμογή προεκπαιδευμένων μοντέλων

Fine Tuning Off-the-shelf Pre-trained Models

awesome-transfer-learning#policy-transfer-for-rl

Προεκπαιδευμένα μοντέλα για εξαγωγή χαρακτηριστικών

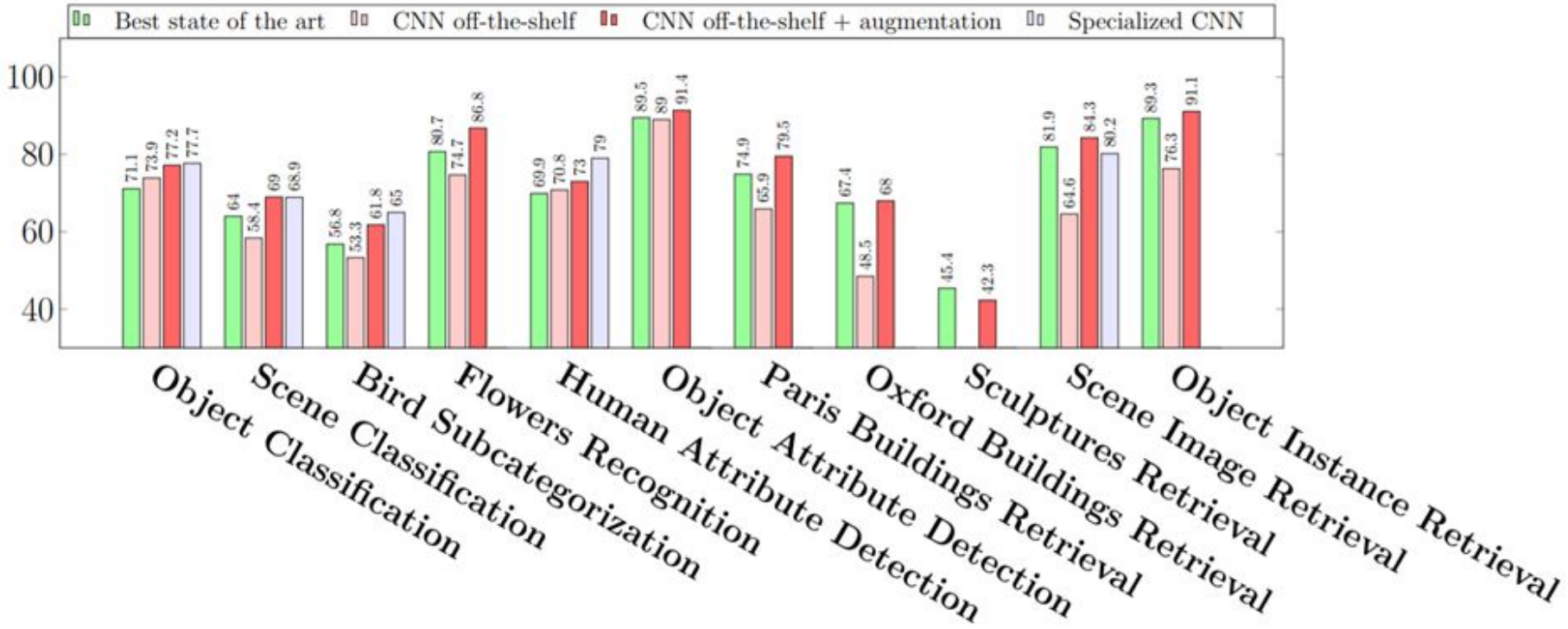
- ➔ Η έξοδο μετά από κάποιο επίπεδο ενός δικτύου βαθιάς μάθησης, που εκπαιδεύτηκε σε διαφορετική εργασίας ($T_s \neq T_t$), χρησιμοποιείται ως γενικευμένος ανιχνευτής χαρακτηριστικών.
- ➔ Εκπαίδευση νέου μοντέλου (π.χ. SVM) με μεταφορά αυτών των χαρακτηριστικών.



Transfer Learning with Pre-trained Deep Learning Models as Feature Extractors

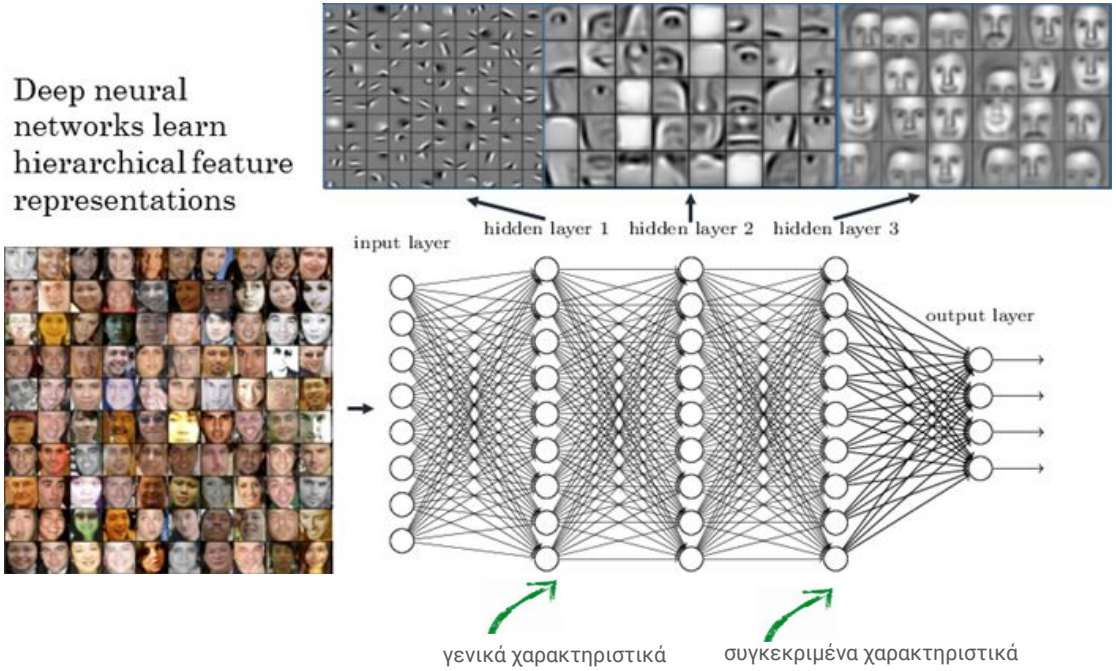
Προεκπαιδευμένα μοντέλα για εξαγωγή χαρακτηριστικών

Τα προ-εκπαιδευμένα χαρακτηριστικά λειτουργούν πολύ καλά για διαφορετικές εργασίες



Βελτίωση προεκπαιδευμένων μοντέλων (Fine tuning)

Δεν αντικαθιστούμε απλώς το τελικό επίπεδο (για ταξινόμηση / παλινδρόμηση), αλλά επανεκπαιδεύουμε επιλεκτικά ορισμένα από τα προηγούμενα επίπεδα.



Βελτίωση προεκπαιδευμένων μοντέλων

Υλοποίηση

- ➔ Παγώνοντας (διορθώνοντας βάρη) ορισμένα επίπεδα του δικτύου κατά την επανεκπαίδευση

- ➔ Τελειοποιώντας τα υπόλοιπα επίπεδα ανάλογα με τις ανάγκες μας.

  Αναλυτικότερα, χρησιμοποιούμε τη γνώση της συνολικής αρχιτεκτονικής του δικτύου και χρησιμοποιούμε τις καταστάσεις του ως σημείο εκκίνησης για το βήμα της επανεκπαίδευσης.

  Αυτό, με τη σειρά του, μας βοηθά να επιτύχουμε καλύτερες επιδόσεις με λιγότερο χρόνο εκπαίδευσης.

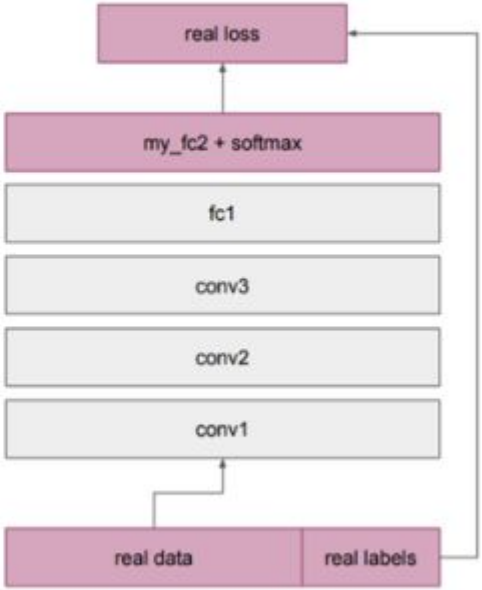
Βελτίωση: Επιβλεπόμενη προσαρμογή τομέα

Στόχος: Ευθυγράμμιση D_s με το D_t

➔ Το δίκτυο εκπαιδεύεται με το T_t να είναι κοντά με το T_s από το οποίο και λαμβάνουμε τις ετικέτες του

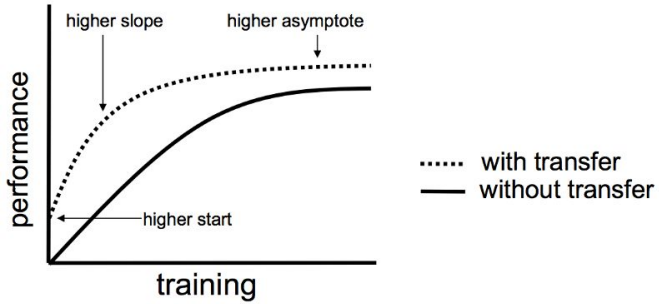
Π.χ. Imagenet classification

➔ “Κόβουμε” τα τελευταία επίπεδα και τα αντικαθιστούμε με πλήρως συνδεδεμένο δίκτυο.



Τα ακριβής προσαρμογής (fine-tuning) δίκτυα χρησιμοποιούν backpropagation με τις ετικέτες του πεδίου προορισμού, μέχρις ότου να αρχίζει να αυξάνει το validation loss.

Κανόνες αξιολόγησης για χρήση TL



Για να αξιολογήσουμε αν ένα μοντέλο μπορεί να μεταφερθεί σε άλλη εργασία πρέπει να λάβουμε υπόψιν μας τα εξής:

Higher start: Η αρχική ικανότητα (πριν τελειοποιήσετε το μοντέλο) στο μοντέλο προέλευσης είναι υψηλότερη από ό, τι διαφορετικά θα ήταν.

Higher slope: Ο ρυθμός βελτίωσης της ικανότητας κατά τη διάρκεια της εκπαίδευσης του μοντέλου προέλευσης είναι πιο απότομος από ότι διαφορετικά θα ήταν.

Higher asymptote : Η συγκλίνουσα ικανότητα του εκπαιδευμένου μοντέλου είναι καλύτερη από ό, τι διαφορετικά θα ήταν.

Πρακτικές συμβουλές

➔ Περιορισμοί από προκατασκευασμένα μοντέλα.

- Η χρήση ενός προκαθορισμένου δίκτυου, ενδέχεται να είναι **δεσμευτική ως προς την αρχιτεκτονική** που μπορείτε να χρησιμοποιήσετε για το νέο σύνολο δεδομένων μας.
  - π.χ. δεν μπορείτε να αφαιρέσετε αυθαίρετα Conv επίπεδα από το προκαθορισμένο δίκτυο.
- Συνήθως χρησιμοποιούμε **μικρότερο learning rate** για τα ρυθμισμένα βάρη ConvNet, σε σύγκριση με τα (τυχαία αρχικοποιημένα) βάρη που θα χρησιμοποιούσαμε για το νέο γραμμικό ταξινομητή που υπολογίζει τα βάρη ταξινόμησης του νέου συνόλου δεδομένων μας.
  - Αυτό συμβαίνει επειδή περιμένουμε ότι τα ρυθμισμένα βάρη ConvNet είναι σχετικά καλά, επομένως δεν θέλουμε να τα παραμορφώσουμε πολύ γρήγορα και πάρα πολύ.

Παραδείγματα Transfer Learning με Deep Learning

Transfer Learning για επεξεργασία εικόνων

Προ-εκπαιδευμένα μοντέλα βαθιάς μάθησης πάνω σε μεγάλα και δύσκολα δεδομένα (π.χ. ImageNet) από ερευνητικές ομάδες με άδεια χρήσης παρέχονται για επαναχρησιμοποίηση σε εργασίες ταξινόμησης εικόνων.

Transfer Learning για επεξεργασία φυσικής γλώσσας

Χρήση προ-εκπαιδευμένων μοντέλων βαθιάς μάθησης για εργασίες επεξεργασίας φυσικής γλώσσας. Τα μοντέλα αυτά είναι εκπαιδευμένα σε πολύ μεγάλο αριθμό εγγράφων κειμένου και παρέχονται για επαναχρησιμοποίηση με άδεια χρήσης.

Transfer Learning για επεξεργασία εικόνων

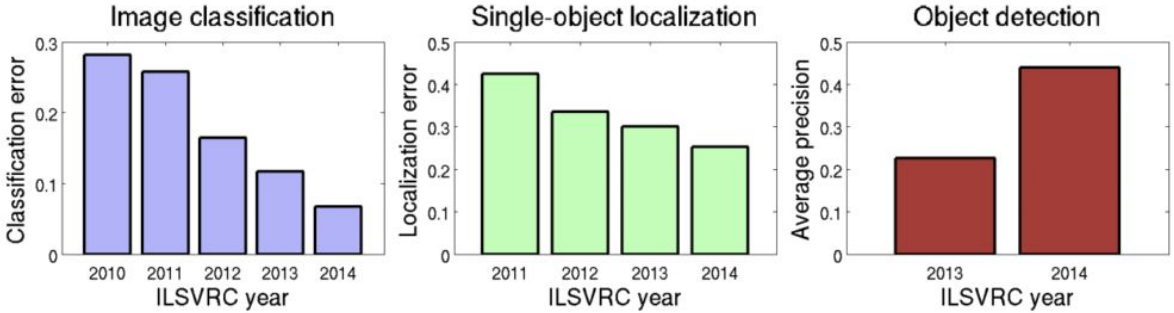
ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

- ILSVRC: ετήσιος διαγωνισμός που χρησιμοποιεί υποσύνολα από το σύνολο δεδομένων ImageNet για ανάπτυξη και συγκριτική αξιολόγηση αλγορίθμων τελευταίας τεχνολογίας.

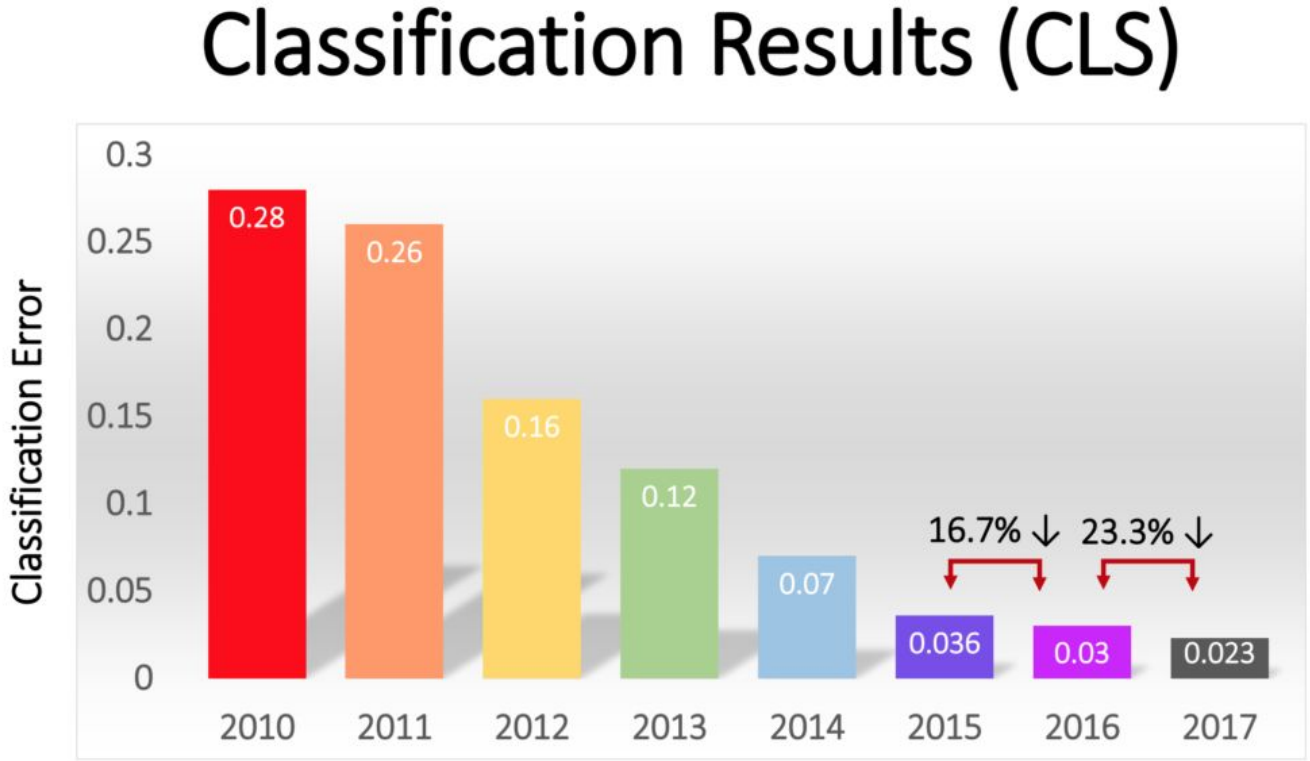
- ImageNet: πολύ μεγάλη συλλογή χαρακτηρισμένων (Amazon Mechanical Turk Worker) φωτογραφιών για την ανάπτυξη αλγορίθμων όρασης υπολογιστή.

- Οι εργασίες του ILSVRC οδήγησαν σε σημαντικές αρχιτεκτονικές μοντέλων και τεχνικές σύνδεσης της όρασης υπολογιστή και της βαθιάς μάθησης

IMAGENET



ImageNet Large Scale Visual Recognition Challenge (ILSVRC)



Eight Years of Competitions

IMAGENET

2010-2017

10× reduction of image classification error

3× improvement of detection precision

What Happens Now?

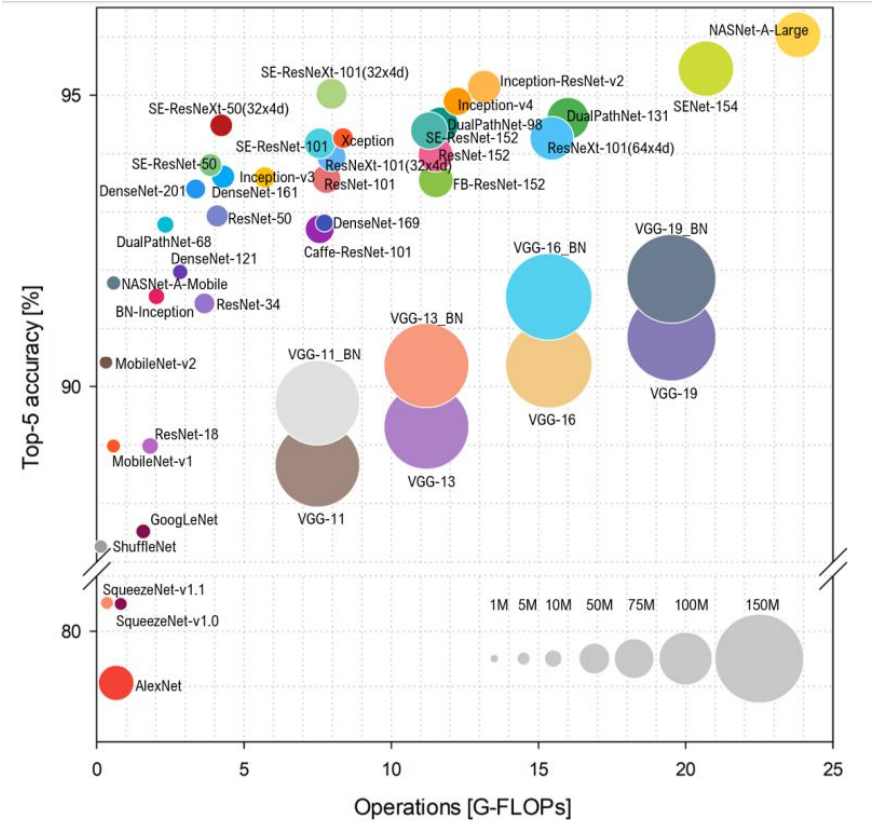
IMAGENET + kaggle

ImageNet **Object Localization** Challenge

ImageNet **Object Detection** Challenge

ImageNet **Object Detection from Video** Challenge

ImageNet Large Scale Visual Recognition Challenge (ILSVRC)



Μοντέλα- Ορόσημα

Καθοριστικοί παράγοντες

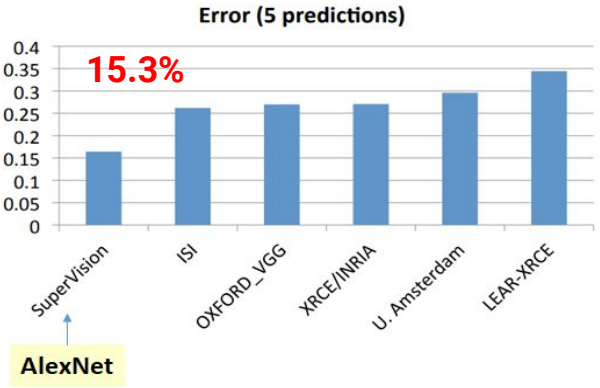
- GPUs
- CNN (ConvNet)

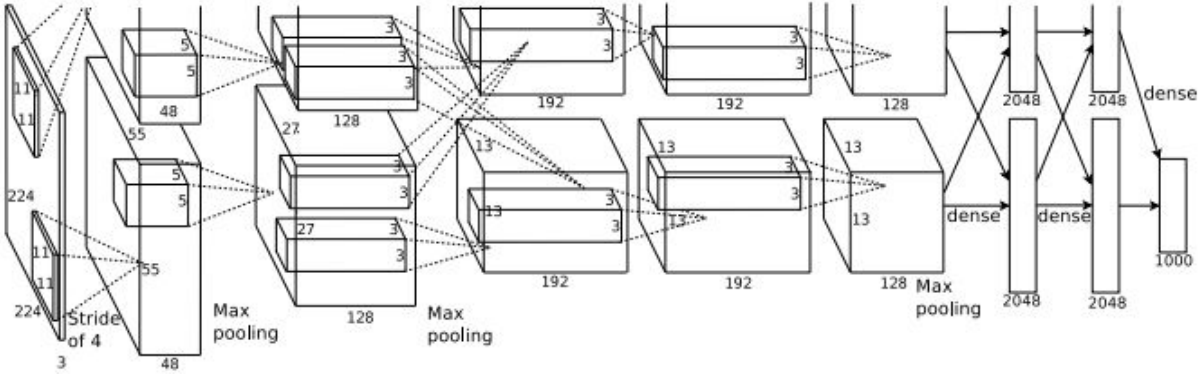
Ένα σύγχρονο ConvNets χρειάζεται 2-3 εβδομάδες για να εκπαιδευτεί σε πολλές GPU στο ImageNet

https://arxiv.org/pdf/1810.00736.pdf

ILSVRC-2012: AlexNet

ImageNet Classification with Deep Convolutional Neural Networks, 2012.
Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton. Univ. of Toronto, Canada





- ➔ Χρήση GPUs κατά τη διάρκεια εκπαίδευσης
- ➔ 8 layers, 5 convolutional and 3 fully-connected, Rectified Linear Units (ReLUs), 60M parameters

ILSVRC-2012: AlexNet

- ➔ Είσοδος: σύνολο εικόνων RGB μεγέθους 227x227x3
- ➔ Έξοδος: διάνυσμα πιθανότητας 1000x1 (ImageNet, 1000 κατηγορίες αντικειμένων)
- ➔ Αύξηση των δεδομένων (κατοπτρισμό, περικοπή)→ μείωση overfitting
- ➔ 3x3 max pooling layer με βήμα 2, μετά από το 1ο, το 2ο και το 5ο Conv layer

- ➔ Αντιμετώπιση του Vanishing Gradient (VG) με ReLU (25% ποσοστό σφάλματος 25%, 6 φορές γρηγορότερο από το ίδιο δίκτυο με tanh)

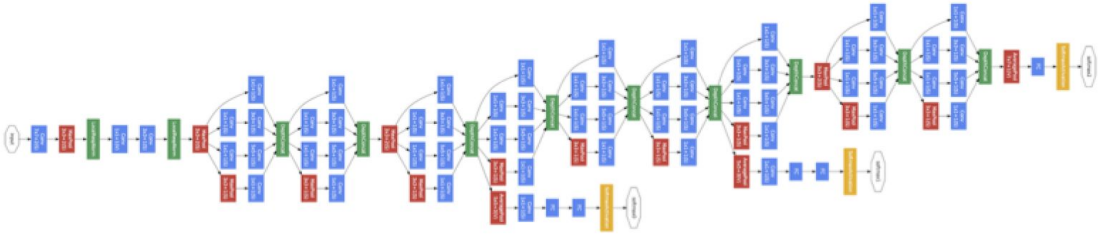
- ➔ Χρησιμοποιεί επίπεδα drop-out με πιθανότητα p = 0.5, αποφεύγοντας τα τοπικά ελάχιστα, αλλά διπλασιάζεται ο αριθμός των επαναλήψεων που απαιτούνται για τη σύγκλιση.

- ➔ Εισήγαγε την ομαλοποίηση τοπικής απόκρισης (LRN): πραγματοποιεί μια ομαλοποίηση σε μια γειτονιά εικονοστοιχείων που ενισχύει τον διεγερμένο νευρώνα ενώ ταυτόχρονα υποβαθμίζει τους γειτονικούς νευρώνες.

ILSVRC-2014: Inception (GoogLeNet)

C. Szegedy *et al.*, "Going deeper with convolutions," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, 2015, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.

➔ Διαδοχικά αλλά και παράλληλα CNN (error rate 6.7%)

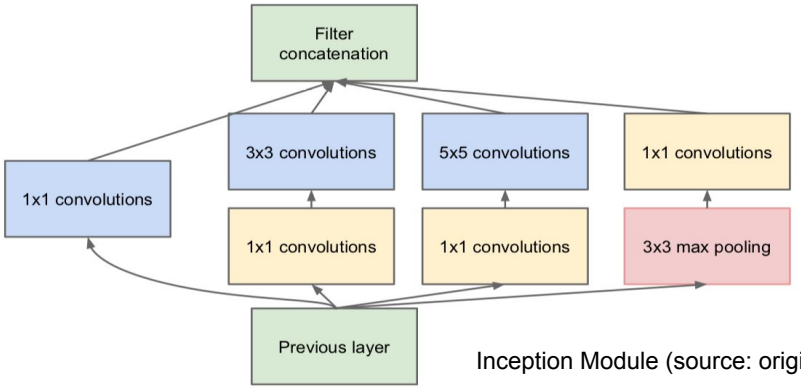


➔ Πολλαπλοί πυρήνες διαφορετικών μεγεθών εφαρμόζονται στο ίδιο επίπεδο με σκοπό τη ανίχνευση συγκεκριμένων χαρακτηριστικών

Περιοχής → κατά βάθος διαχωρίσιμη συνελιξη (Depthwise separable convolution)

- Μεγάλοι πυρήνες → καθολικά χαρακτηριστικά που κατανέμονται σε μεγάλη περιοχή της εικόνας,
- Μικροί πυρήνες → ανίχνευση συγκεκριμένων χαρακτηριστικών περιοχής που κατανέμονται σε ολόκληρο το πλαίσιο εικόνας.

ILSVRC-2014: Inception (

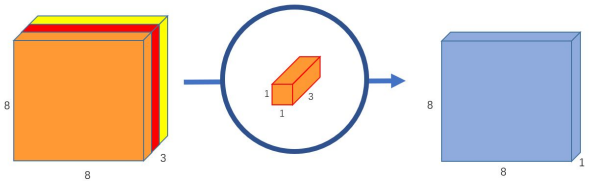


Inception Module (source: original paper)

Μονάδα Inception :

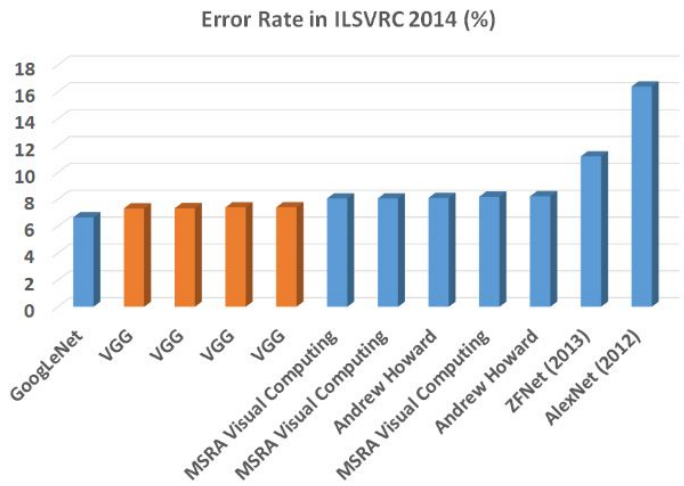
Καταγράφει προεξέχοντα χαρακτηριστικά (salient features) σε διαφορετικά επίπεδα.

- ➔ 4 παράλληλες λειτουργίες
  - 1x1 conv layer, μείωση βάθους (pointwise conv)
  - 3x3 conv layer, Κατανεμημένα χαρακτηριστικά (distributed features)
  - 5x5 conv layer, Γενικά χαρακτηριστικά (global features)
  - max pooling, Χαμηλού επιπέδου χαρακτηριστικά (low level features)



- ➔ Φίλτρο συνένωσης
  π.χ. εάν οι εικόνες στο σύνολο δεδομένων έχουν πολλά καθολικά χαρακτηριστικά και ελάχιστα χαρακτηριστικά χαμηλού επιπέδου, τότε το εκπαιδευμένο δίκτυο Inception θα έχει πολύ μικρά βάρη που αντιστοιχούν στον πυρήνα 3x3 σε σύγκριση με τον πυρήνα 5x5.

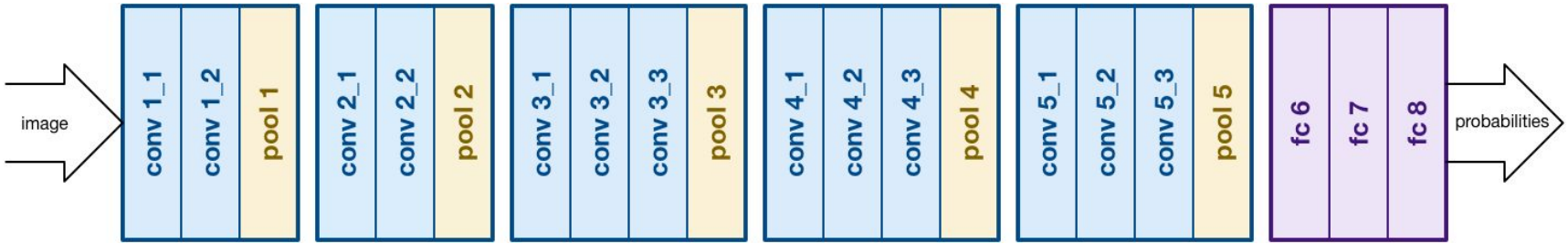
ILSVRC-2014: VGG



[Simonyan, K. and Zisserman, A. (2015) Very Deep Convolutional Networks for Large-Scale Image Recognition. 3rd International Conference on Learning Representations (ICLR2015).]

13 συνελικτικά και 3 πλήρως συνδεδεμένα επίπεδα, ReLU, φίλτρα μικρότερου μεγέθους (2 × 2 και 3 × 3) από το AlexNet , 138M παραμέτρους, 500MB

- ➔ Μείωση του αριθμού των παραμέτρων στα επίπεδα CONV
- ➔ Βελτίωση του χρόνου εκπαίδευσης
- ➔ Σχεδίασαν επίσης βαθύτερες παραλλαγές, VGG-16, VGG-19.



ILSVRC-2014: VGG-16

Η ιδέα πίσω από την ύπαρξη πυρήνων σταθερού μεγέθους είναι ότι όλοι οι conv πυρήνες μεταβλητού μεγέθους που χρησιμοποιούνται στο Alexnet (11x11, 5x5, 3x3) μπορούν να αναπαραχθούν χρησιμοποιώντας πολλαπλούς πυρήνες 3x3 ως δομικά στοιχεία.

π.χ. Έστω επίπεδο εισόδου μεγέθους 5x5x1

Περίπτωση 1: 1ο conv επίπεδο: ένας πυρήνας 5x5 και βήμα 1 → Έξοδος: χάρτης χαρακτηριστικών 1x1

Πλήθος μεταβλητών 5x5x1=25 ((m*n+1)*k, k: πλήθος πυρήνων)

Περίπτωση 2: 1ο conv επίπεδο: δύο πυρήνες 3x3 και βήμα 1 → Έξοδος: χάρτης χαρακτηριστικών 1x1.

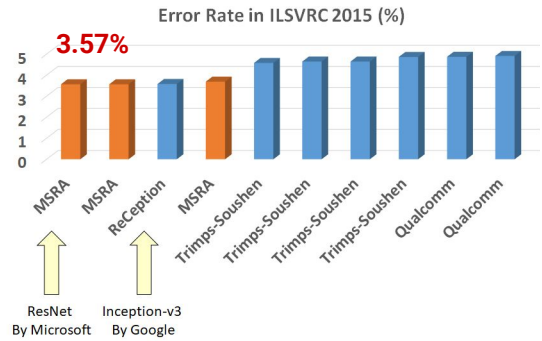
Πλήθος μεταβλητών 3x3x2=18 → Μείωση 28%

Αντίστοιχα αντί για χρήση πυρήνων 7x7 (11x11) εφαρμόσουμε 3 (5) 3x3 πυρήνες → μείωση αριθμού εκπαιδευόμενων μεταβλητών κατά 44,9% (62,8%)

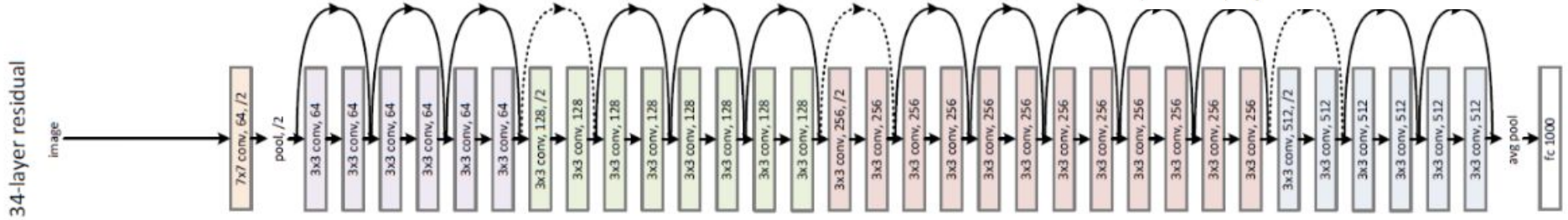
★ Ταχύτερη εκμάθηση

★ Αποφυγή overfitting

ILSVRC-2015: ResNet (MSRA)



Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun Deep Residual Learning for Image Recognition, CVPR 2015



➔ 152 επίπεδα, 11M παράμετροι, πυρήνες, 3x3 (όπως το VGGNet), 2 pooling επίπεδα

Σύνδεση ταυτότητας (Identity connection) ανά δύο επιπέδων CONV, διάσταση εισόδου ίδια με της εξόδου

Σύνδεση προβολής (Projection connection) όπου οι διαστάσεις εισόδου διαφέρουν με της εξόδου.

Υπάρχουν πολλές εκδόσεις αρχιτεκτονικών ResNetXX όπου το «XX» υποδηλώνει τον αριθμό των επιπέδων (ResNet50, ResNet101)

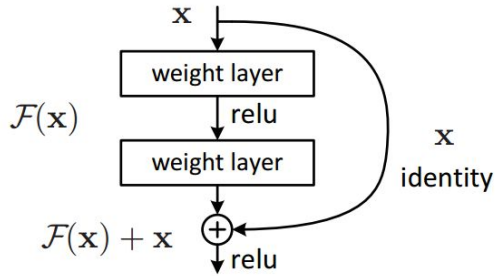
ILSVRC-2015: ResNet (MSRA)

Αντί να μάθει την αναπαράσταση x → F (x), το δίκτυο μαθαίνει την x → F (x) + G (x)

➔ Όταν x=F (x), η συνάρτηση G (x) = x είναι μια συνάρτηση ταυτότητας → σύνδεση ταυτότητας
Η αναπαράσταση όταν x=F (x) μαθαίνεται με μηδενισμό των βαρών στο ενδιάμεσο στρώμα κατά τη διάρκεια της εκπαίδευσης

➔ Όταν x≠F (x) (stride> 1 στα μεταξύ τους επίπεδα CONV) → σύνδεση προβολής
Η συνάρτηση G (x) αλλάζει τις διαστάσεις της εισόδου x
σε εκείνη της εξόδου F (x)

residual block



Είδη αντιστοίχισης

- Non-trainable Mapping (Padding)
  Η είσοδος x είναι απλώς γεμάτη με μηδενικά για να ταιριάζει η διάσταση με εκείνη του F (x).
- Trainable Mapping (Conv Layer)
  1x1 Conv layer χρησιμοποιείται για την αντιστοίχιση του x με την G (x).

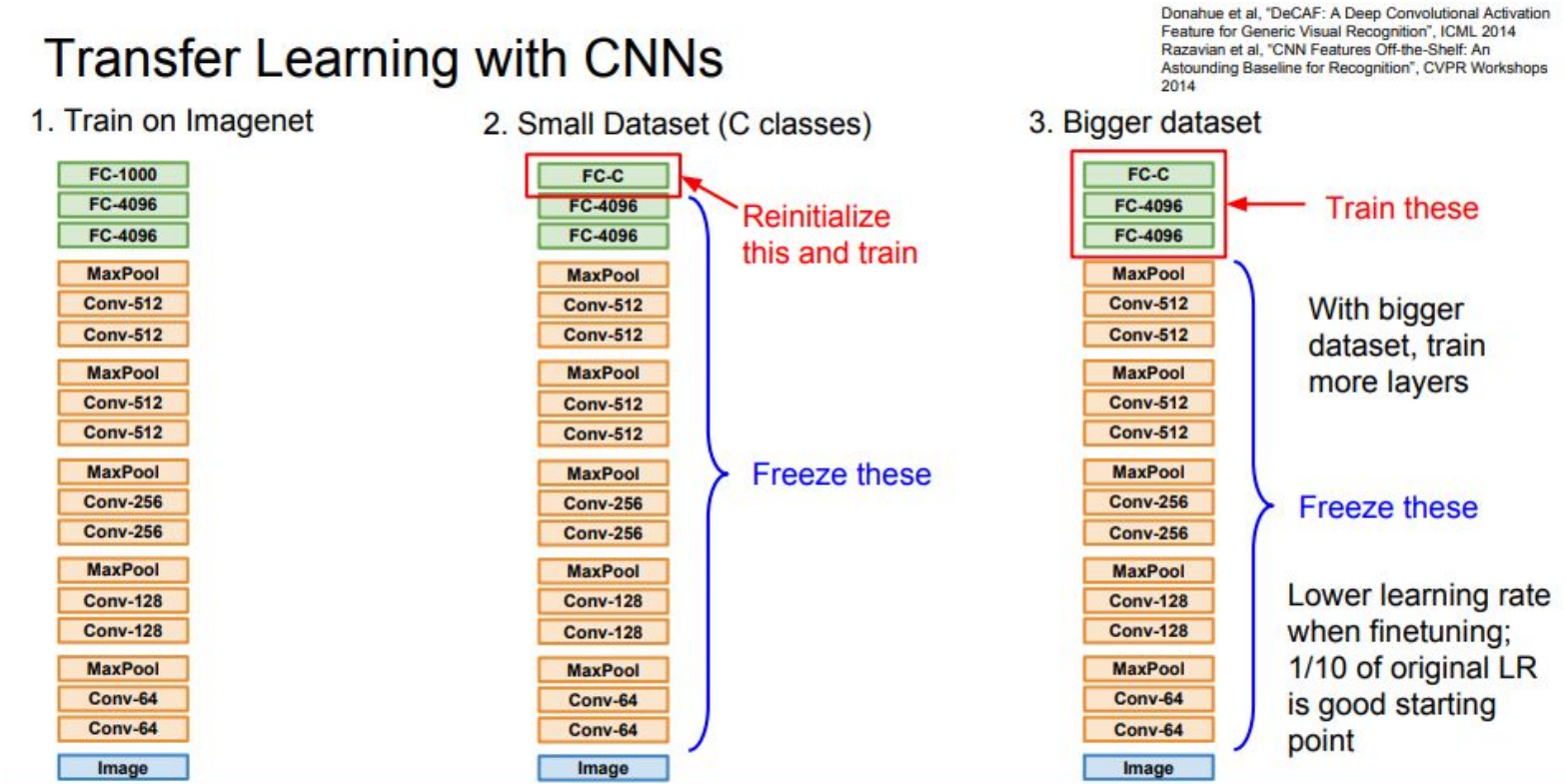
Παράδειγμα: ConvNet για εξαγωγή χαρακτηριστικών

- ➔ Πάρτε ένα προκατασκευασμένο ConvNet στο ImageNet
- ➔ Αφαιρέστε το τελευταίο πλήρως συνδεδεμένο επίπεδο (οι έξοδοι αυτού του επιπέδου είναι οι βαθμολογίες 1000 τάξεων του ImageNet)
- ➔ Αντιμετωπίστε το υπόλοιπο του ConvNet ως εξαγωγέα χαρακτηριστικού για το νέο σύνολο δεδομένων

π.χ. Σε ένα AlexNet, υπολογίζεται ένα διάνυσμα 4096-D για κάθε εικόνα που περιέχει τις ενεργοποιήσεις του κρυφού επιπέδου αμέσως πριν τον ταξινομητή.

Μόλις εξαχθούν τα διανύσματα 4096-D για όλες τις εικόνες, μπορούμε να εκπαιδεύσουμε έναν γραμμικό ταξινομητή (π.χ. Linear SVM ή Softmax classifier) για το νέο σύνολο δεδομένων.

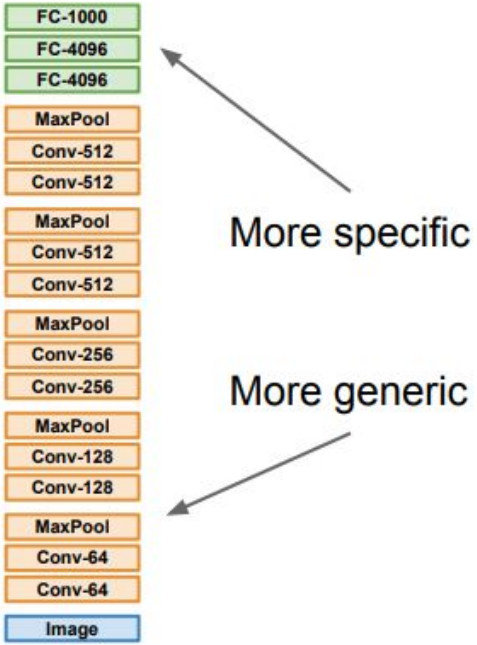
Παράδειγμα:ConvNet για εξαγωγή χαρακτηριστικών



Βελτίωση του ConvNet.

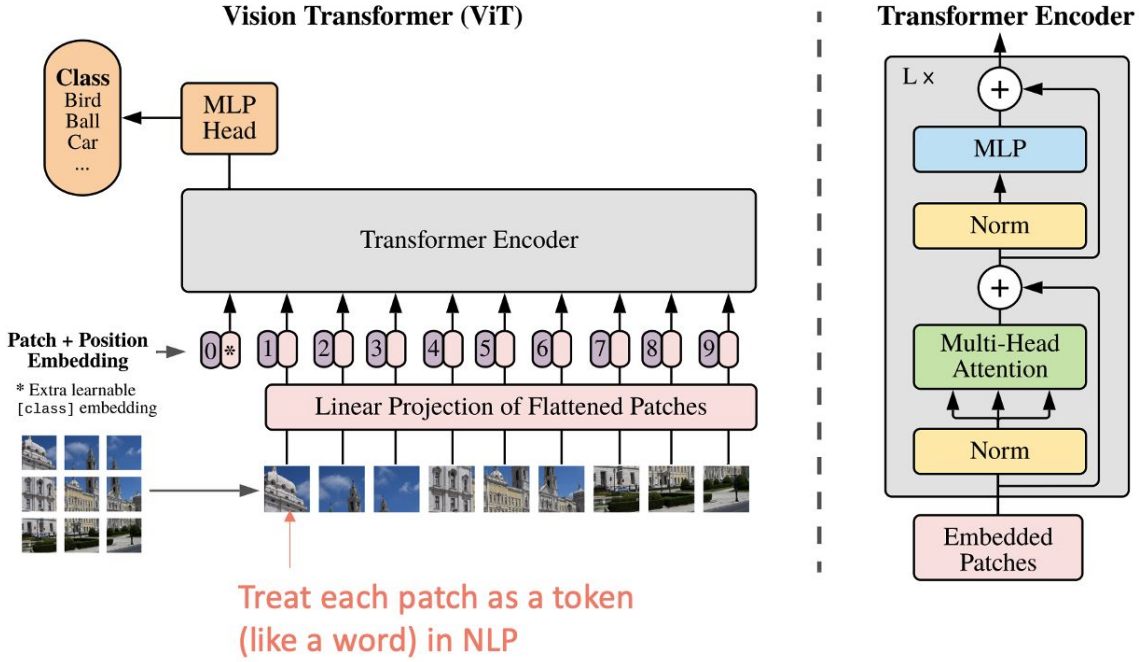
- Πάρτε ένα προκατασκευασμένο ConvNet στο ImageNet
- Αντικαταστήστε και επανεκπαιδεύστε τον ταξινομητή πάνω από το ConvNet στο νέο σύνολο δεδομένων,
- Βελτιώσετε τα βάρη του προκατασκευασμένου δικτύου συνεχίζοντας το backpropagation
- Είναι δυνατό να τελειοποιήσετε όλα τα επίπεδα του ConvNet ή είναι δυνατόν να διατηρήσετε ορισμένα από τα προηγούμενα επίπεδα σταθερά (ώστε να αποφύγετε υπερφόρτωση) και μόνο να βελτιώσετε τα υψηλότερα επίπεδα του δικτύου.

Πρακτικές συμβουλές βελτίωσης



	very similar dataset	very different dataset
very little data	Use Linear Classifier on top layer	You're in trouble… Try linear classifier from different stages
quite a lot of data	Finetune a few layers	Finetune a larger number of layers

Vision Transformer (ViT)



Το Vision Transformer (ViT) είναι ένα μοντέλο που εφαρμόζει το Transformer στην εργασία ταξινόμησης εικόνων και προτάθηκε τον Οκτώβριο του 2020 (Dosovitskiy et al. 2020).

[An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale]

Vision Transformer Architecture

1. Διαχωρίστε μια εικόνα σε patches
2. Μετατρέψτε τα patches σε μονοδιάστατα διανύσματα (flattened patches)
3. Παράξτε lower-dimensional linear embeddings από τα flattened patches
4. Προσθέστε positional embeddings
5. Εισάγετέ την ως είσοδο στον standard transformer encoder
6. Προεκπαίδευση του μοντέλου με ετικέτες εικόνας (πλήρης επίβλεψη σε ένα τεράστιο σύνολο δεδομένων)
7. Finetune χρησιμοποιώντας κάποιο dataset για image classification

Προεκπαιδευμένα μοντέλα

- Tensor Flow Hub
- PyTorch Hub
- Model Zoo · BVLC/caffe Wiki · GitHub
- tensorflow/models: Models and examples built with TensorFlow
- Keras Applications

Υλοποιήσεις για CV

- https://keras.io/guides/transfer_learning/
- hands-on-transfer-learning-with-python/CIFAR10_VGG16_TLClassifier
- pytorch/vision: Datasets, Transforms and Models specific to Computer Vision
- Detecto — An object detection library for PyTorch - PyTorch
- https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html
- https://www.tensorflow.org/tutorials/images/transfer_learning_with_hub?hl=da
- https://www.tensorflow.org/tutorials/images/transfer_learning?hl=da
- GAN: https://www.tensorflow.org/tutorials/generative/style_transfer?hl=da
- 4. Transfer Learning with Your Own Image Dataset — gluoncv 0.8.0

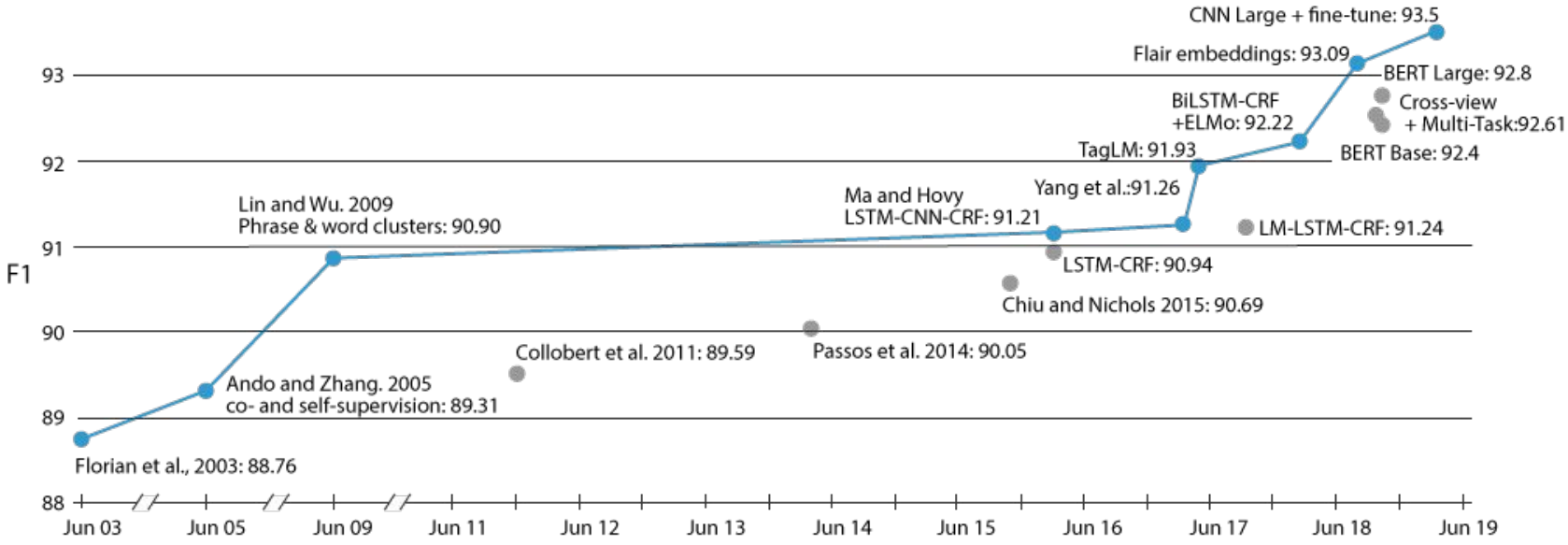
Transfer Learning για επεξεργασία φυσικής γλώσσας

Transfer Learning in Natural Language Processing NAACL-HLT 2019

Χρήση transfer learning σε NLP

Διαμοιρασμός γλωσσικών αναπαραστάσεων , δομικών ομοιοτήτων, σύνταξη, σημασιολογία

Το transfer learning οδήγησε το SOTA στο NLP (π.χ. ταξινόμηση, εξαγωγή πληροφοριών, Q&A,..)



Performance on Named Entity Recognition (NER) on CoNLL-2003 (English)

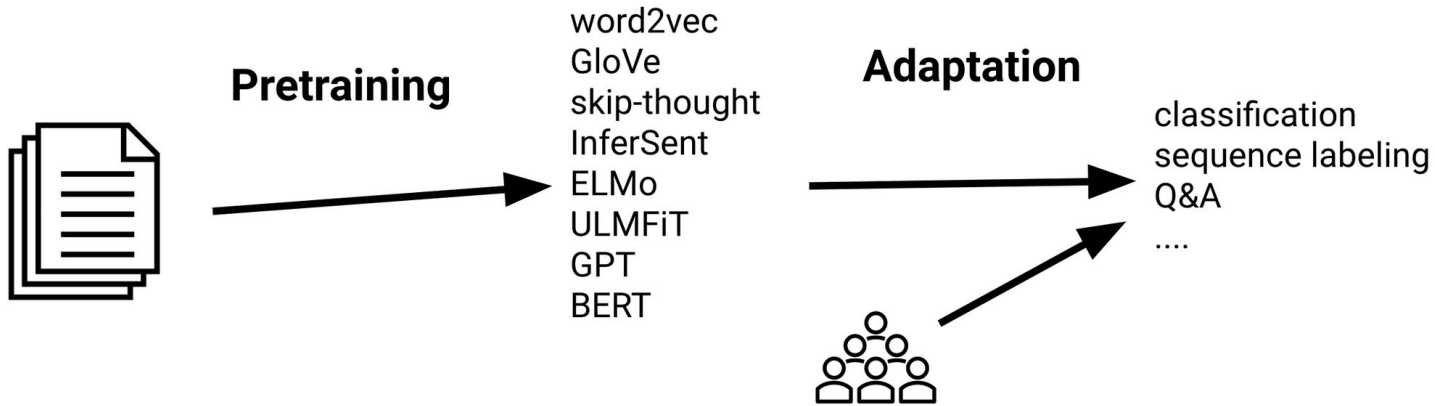
Sequential transfer learning

- Μαθαίνει σε μια εργασία / σύνολο δεδομένων

  Προετοιμάζει τις αναπαραστάσεις σε ένα μεγάλο σώμα κειμένων χωρίς ετικέτα, επιλέγοντας μία μέθοδο.

- Μεταφέρει σε μια άλλη εργασία / σύνολο δεδομένων

  Προσαρμόζει αυτές τις αναπαραστάσεις σε μια εποπτευόμενη εργασία στόχου, χρησιμοποιώντας επισημασμένα δεδομένα.



Sequential transfer learning: Κύρια θέματα

Words to words-in-context: Ενσωμάτωση αναπαραστάσεων σε πλαίσιο

word2vec (Mikolov et al., 2013)

Μαθαίνει μια μοναδική αναπαράσταση για κάθε λέξη ανεξάρτητη από το περιβάλλον της

- σε προτάσεις και κείμενα (Le and Mikolov, 2014; Conneau et al., 2017).
- μαθαίνουν αναπαραστάσεις λέξεων που αλλάζουν με βάση τη σημασιολογία της λέξης (McCann et al., 2017; Peters et al., 2018)

Language modelling pretraining

Εκμάθηση λογικών μοντέλων γλωσσών από μεγάλα κειμενικά δεδομένα

- Εκμάθηση των παραστάσεων φράσεων και λέξεων με μια ποικιλία αντικειμενικών λειτουργιών.

Βαθιά δίκτυα

π.χ. BERT-Large, GPT-2 (24 Transformer block)

NLP

Statistical Machine Translation (SMT)

Κυρίαρχο μοντέλο μετάφρασης για χρόνια,

Neural Machine Translation (NMT)

Δημιουργία και εκπαίδευση ενιαίου, μεγάλου νευρωνικού δικτύου
Είσοδος: κείμενο Έξοδος: μετάφραση κειμένου

(Kalchbrenner και Blunsom (2013), Sutskever et. al (2014) και Cho. κ.ά. al (2014b))

Sutskever et. al (2014) : Εκμάθηση από αλληλουχία σε αλληλουχία (seq2seq)

Attention

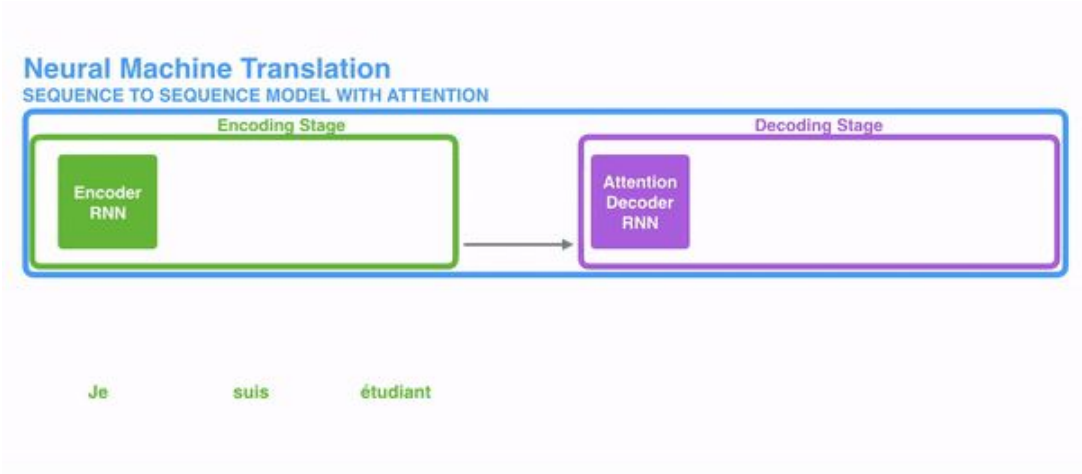
Χρησιμοποιώντας το μηχανισμό του attention, ένα σύστημα μπορεί να εστιάσει σε μέρος ενός υποσυνόλου των πληροφοριών που τους δίνονται.

Η ιδέα πίσω από αυτό είναι ότι μπορεί να υπάρχουν σχετικές πληροφορίες για κάθε λέξη σε μια πρόταση. Έτσι, για να είναι ακριβής η αποκωδικοποίηση, πρέπει να λαμβάνεται υπόψη κάθε λέξη της εισόδου, με το μηχανισμό attention.

Εφαρμογές με attention

machine translation, text summarization, image captioning, dialogue generation, sentiment analysis.

Έχουν προταθεί διάφορες μορφές και τύποι μηχανισμών attention και εξακολουθούν να αποτελούν σημαντικό ερευνητικό πεδίο του NLP



Visualizing machine learning one concept at a time.

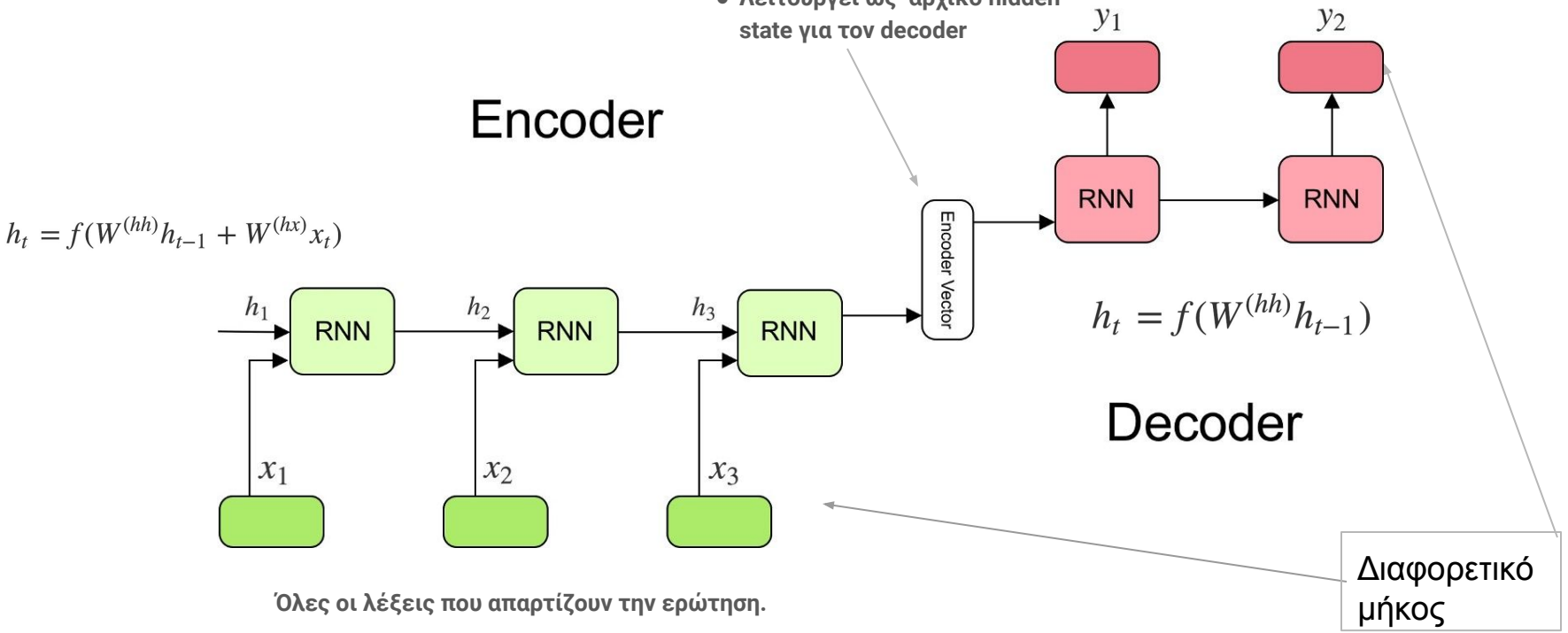
seq2seq + attention model

π.χ. Πρόβλημα question-answering

- Vector με ενσωματωμένες τις πληροφορίες εισόδου.
- Λειτουργεί ως αρχικό hidden state για τον decoder

Δημιουργία ενός διανύσματος πιθανότητας που θα μας βοηθήσει να προσδιορίσουμε την τελική έξοδο

$$y_t = softmax(W^S h_t)$$



https://nlp.stanford.edu/~johnhew/public/14-seq2seq.pdf

seq2seq+attention: Hidden states

```
decoder_hidden = [10, 5, 10]

encoder_hidden
------------------
     [0, 1, 1]
     [5, 0, 1]
     [1, 1, 0]
     [0, 5, 1]
```

➔ Διαθέσιμα encoder hidden states (memory vector, πράσινο)

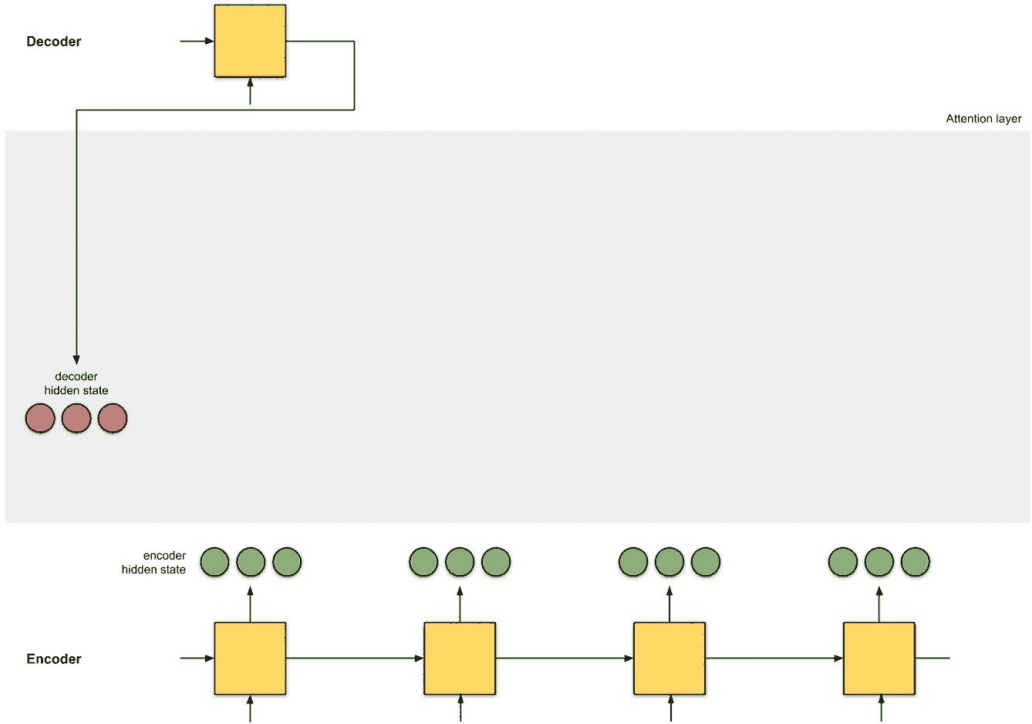
➔ 1ο decoder hidden state (κόκκινο - η τελευταία κωδικοποιημένη encoder hidden state)

Το RNN function, συγκρίνει το word vector με την προηγούμενη state και δημιουργούν τη νέα state

4 “word vectors” στην είσοδο: Διαβάζει 1-1 τις λέξεις εισόδου

https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

seq2seq+attention: σκορ για κάθε encoder hidden state

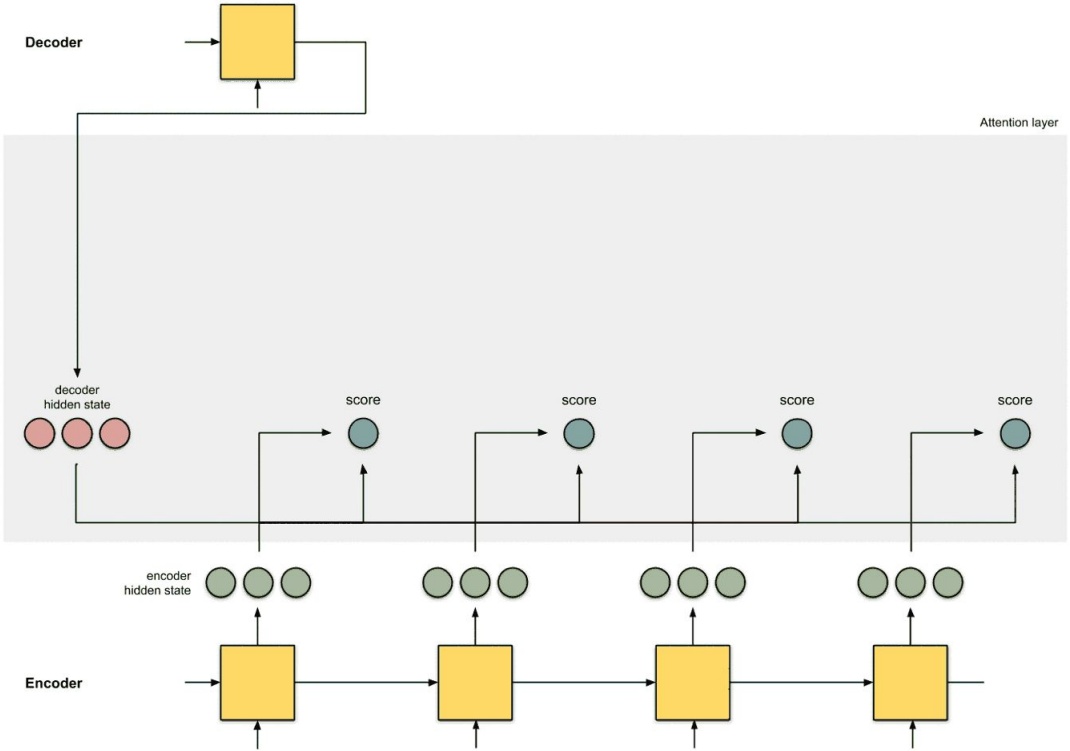


Το attention layer εστιάζει διαφορετικά σε διαφορετικές λέξεις, εκχωρώντας μία βαθμολογία για κάθε λέξη.

```
decoder_hidden = [10, 5, 10]

encoder_hidden  score
----------------------
     [0, 1, 1]     15 (= 10×0 + 5×1 + 10×1, dot product)
     [5, 0, 1]     60  ( high attention score)
     [1, 1, 0]     15
     [0, 5, 1]     35
```

seq2seq+attention: επίπεδο softmax

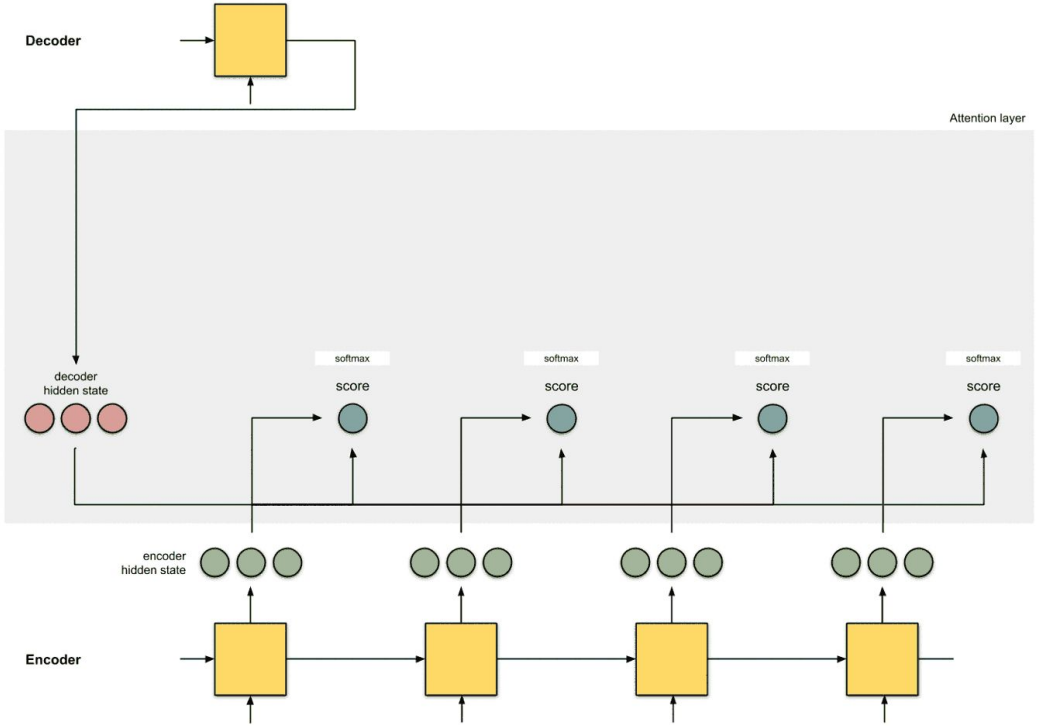


Με ένα επίπεδο softmax κάνουμε κανονικοποίηση των σκορς και οι τιμές που προκύπτουν αποτελούν την κατανομή του attention (attention distribution)

encoder_hidden	score	score^
[0, 1, 1]	15	0
[5, 0, 1]	60	**1**
[1, 1, 0]	15	0
[0, 5, 1]	35	0

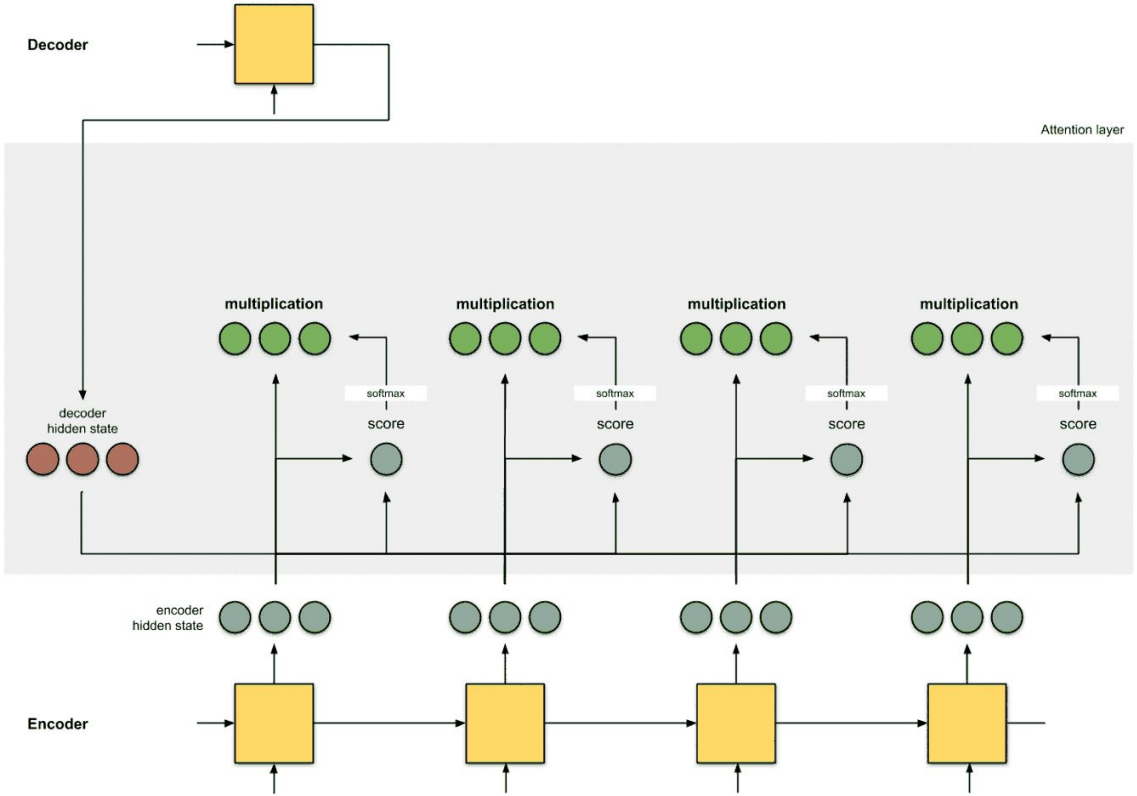
Σε πραγματικό παράδειγμα το score^ μπορεί να πάρει οποιαδήποτε τιμή μεταξύ 0 και 1

seq2seq+attention: Πολλαπλασιασμός κάθε encoder hidden state με το κανονικοποιημένο σκορ



encoder	score	score^	alignment
[0, 1, 1]	15	0	[0, 0, 0]
[5, 0, 1]	60	1	[5, 0, 1]
[1, 1, 0]	15	0	[0, 0, 0]
[0, 5, 1]	35	0	[0, 0, 0]

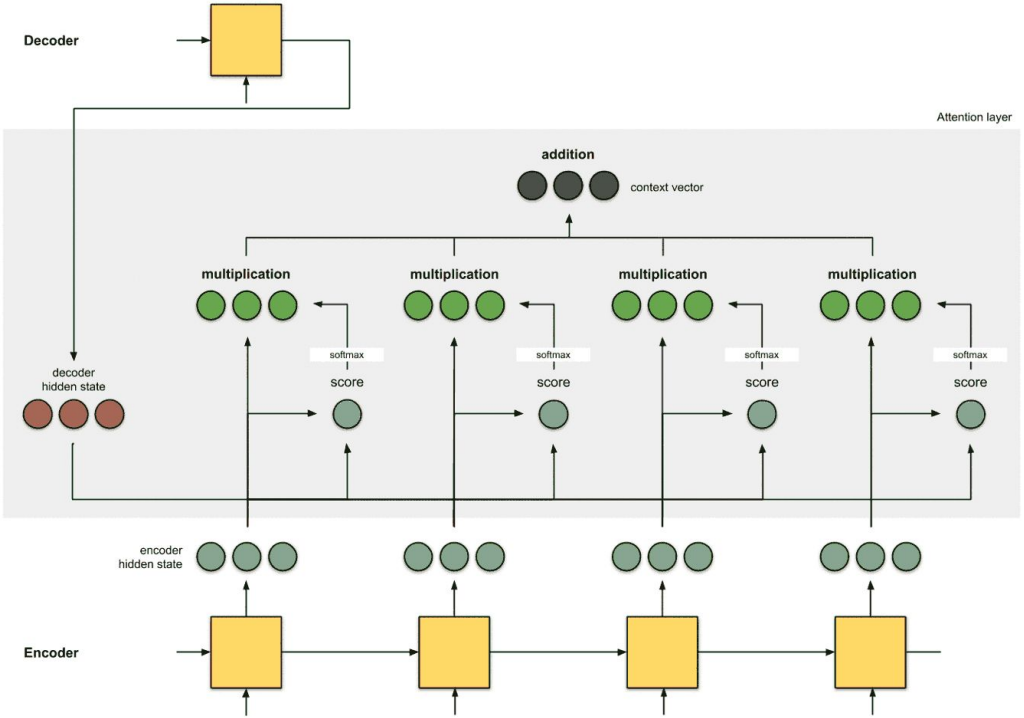
seq2seq+attention: παραγωγή context vector




```
encoder  score  score^  alignment
---------------------------------
[0, 1, 1]   15      0   [0, 0, 0]
[5, 0, 1]   60      1   [5, 0, 1]
[1, 1, 0]   15      0   [0, 0, 0]
[0, 5, 1]   35      0   [0, 0, 0]
```

context = [0+5+0+0, 0+0+0+0, 0+1+0+0] = [5, 0, 1]

seq2seq+attention: τροφοδότηση του context vector στο decoder



Το backpropagation, αλλάζοντας τα βάρη στα RNN και στη συνάρτηση σκορ, θα επηρεάσει τις κρυφές καταστάσεις του κωδικοποιητή και του αποκωδικοποιητή, οι οποίες με τη σειρά τους επηρεάζουν τα αποτελέσματα του attention

2018: NLP's ImageNet moment

http://jalammar.github.io/illustrated-bert/

BERT: Bidirectional Encoder Representations from Transformers

Το BERT είναι ένα μοντέλο αναπαράστασης γλώσσας με εντυπωσιακή ακρίβεια σε πολλές εφαρμογές NLP.

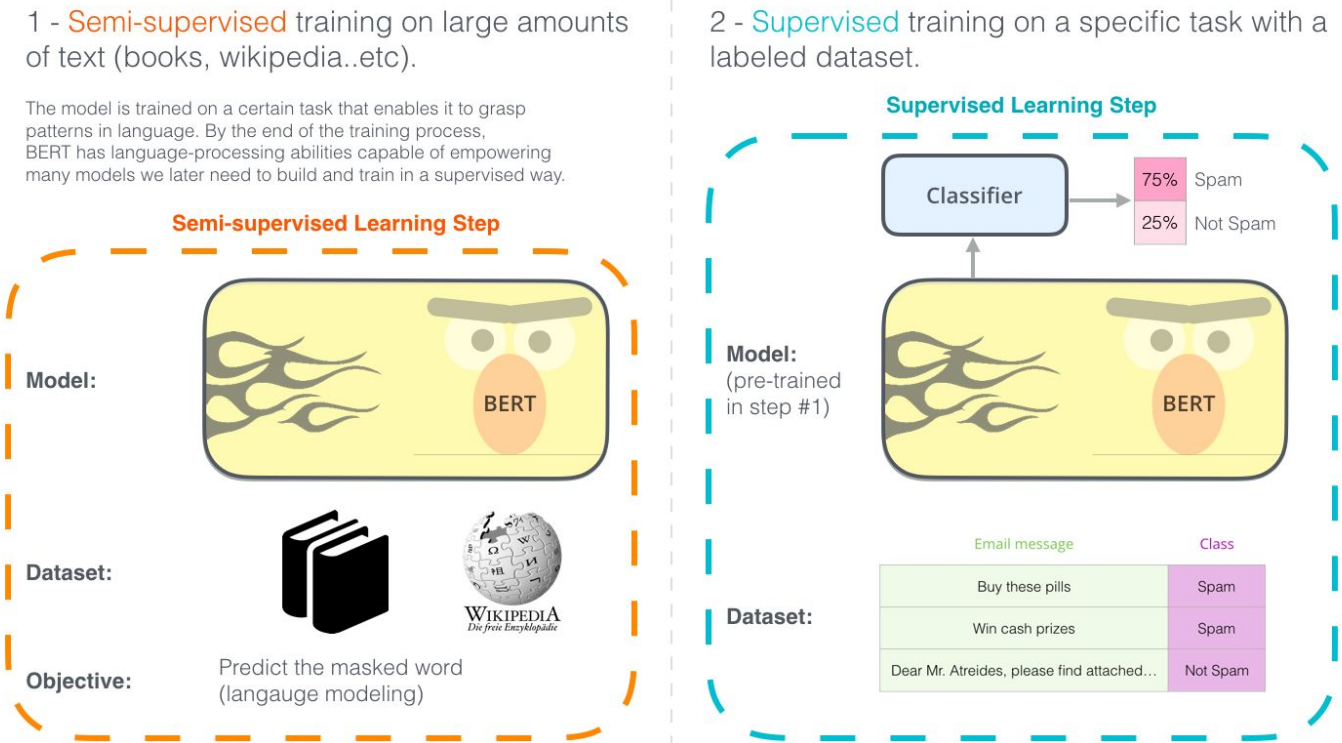
Πρόβλημα

- Σύμφωνα με την Google το 15% των αναζητήσεων που γίνονται καθημερινά στη μηχανή αναζήτησης δεν έχουν πραγματοποιηθεί ξανά.
- Επομένως, το πρόβλημα που εγείρεται δεν αφορά τις ερωτήσεις που μπορεί να κάνουν οι χρήστες αλλά με τους πόσους διαφορετικούς τρόπους μπορεί να γίνει μία αναζήτηση.

Λύση

- Για την αντιμετώπιση αυτού η Google απομακρύνεται από την χρήση λέξεων-κλειδιά για την κατανόηση των αναζητήσεων και τη προχωρά στη χρήση πιο σύνθετων μοντέλων όπως το BERT

BERT: Bidirectional Encoder Representations from Transformers

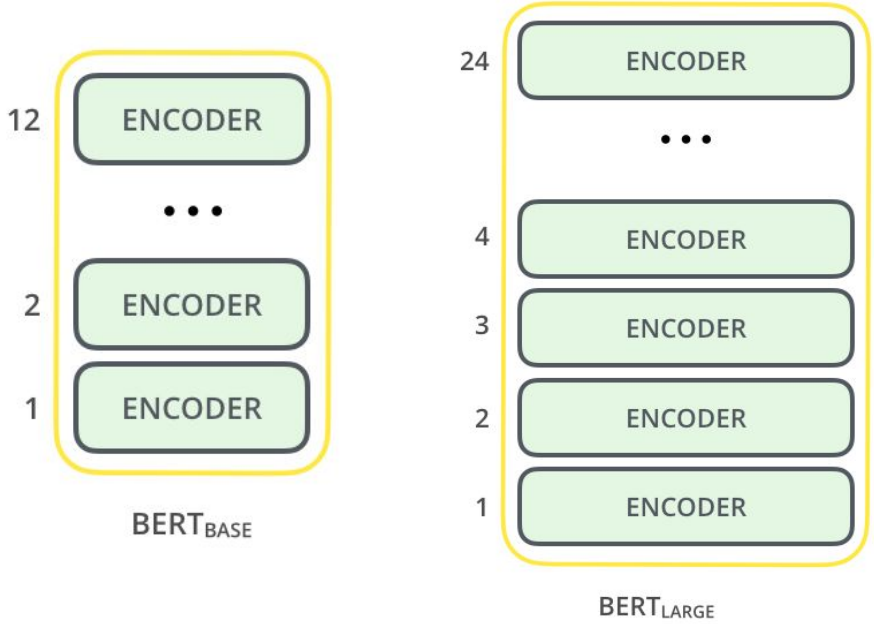


[1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

google-research/bert: TensorFlow code and pre-trained models for BERT

Τα δύο βήματα για τον τρόπο ανάπτυξης του BERT. Μπορείτε να κατεβάσετε το μοντέλο που έχει προ-εκπαιδευτεί στο βήμα 1 (εκπαιδευμένο σε δεδομένα χωρίς σχόλια) και να το βελτιστοποιήσετε στο βήμα 2.

Μοντέλα BERT



	Base version	Large version
encoder layers (Transformer Blocks)	12	24
feedforward-networks	768 hidden units	1024 hidden units
attention heads	12	16

BERT BASE - Συγκρίσιμο σε μέγεθος με το OpenAI Transformer για σύγκριση απόδοσης

BERT LARGE - Ένα τεράστιο μοντέλο που πέτυχε τα αποτελέσματα του paper

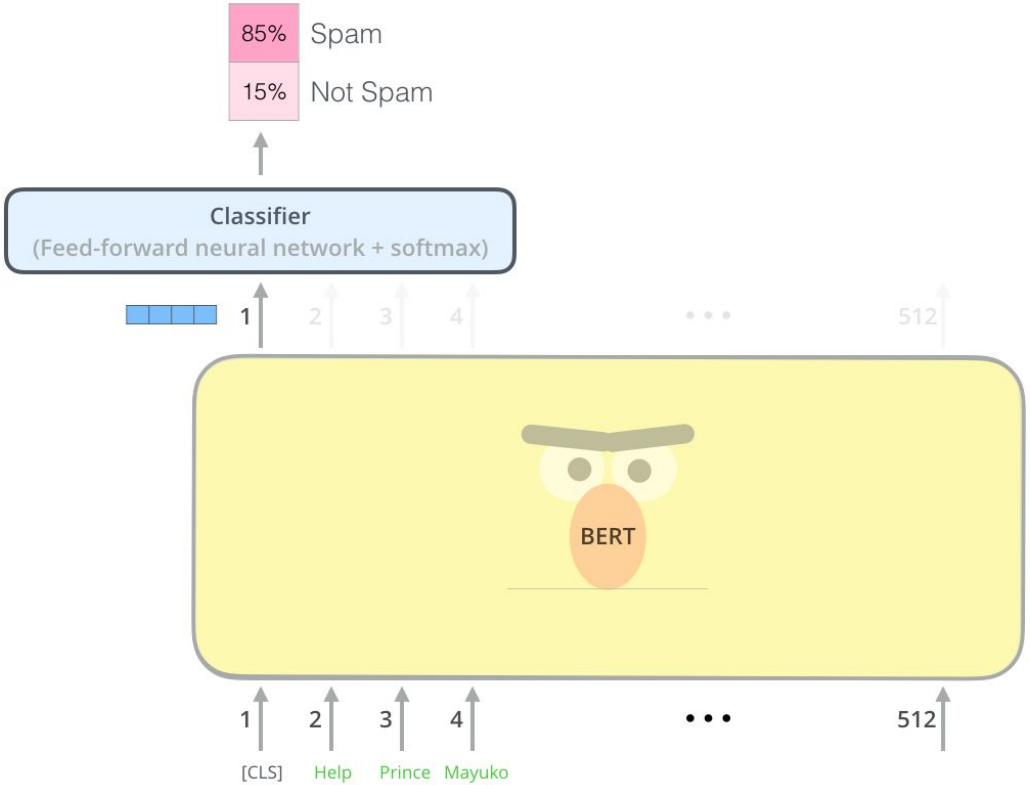
BERT: Bidirectional Encoder Representations from Transformers

- Το σημείο κλειδί στην τεχνική καινοτομία του BERT είναι η ικανότητα του να εφαρμόσει την αμφίδρομη εκπαίδευση των Transformers και έναν δημοφιλή μηχανισμό προσοχής στη μοντελοποίηση της φυσικής γλώσσας.

- Αυτό έρχεται σε αντίθεση με προηγούμενες προσπάθειες που εξέταζαν μία ακολουθία κειμένου από τα αριστερά προς τα δεξιά (και αντιστρόφως).

- Η τεχνική Masked LM που εφάρμοσε το BERT έδειξε ότι ένα μοντέλο φυσικής γλώσσας που είναι αμφίδρομα εκπαιδευμένο μπορεί να αποκτήσει καλύτερη κατανόηση για το περιεχόμενο ενός κειμένου και την ροή του λόγου σε σύγκριση με ένα μοντέλο μονής κατεύθυνσης

BERT: Bidirectional Encoder Representations from Transformers



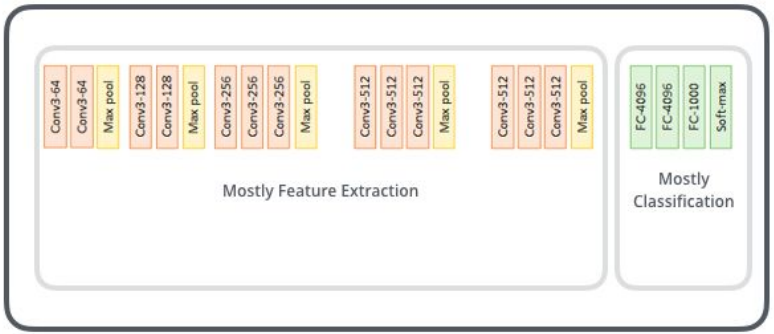
Input: sequence of words

Παραλληλισμός με τα ConvNets

Input

Features

VGG-16



Output

Prediction

0.2%	Kit fox
0.1%	English setter
95%	Egyptian cat
1%	Great Dane
	…
0%	Hotdog

BERT: από τους Decoders στους Encoders

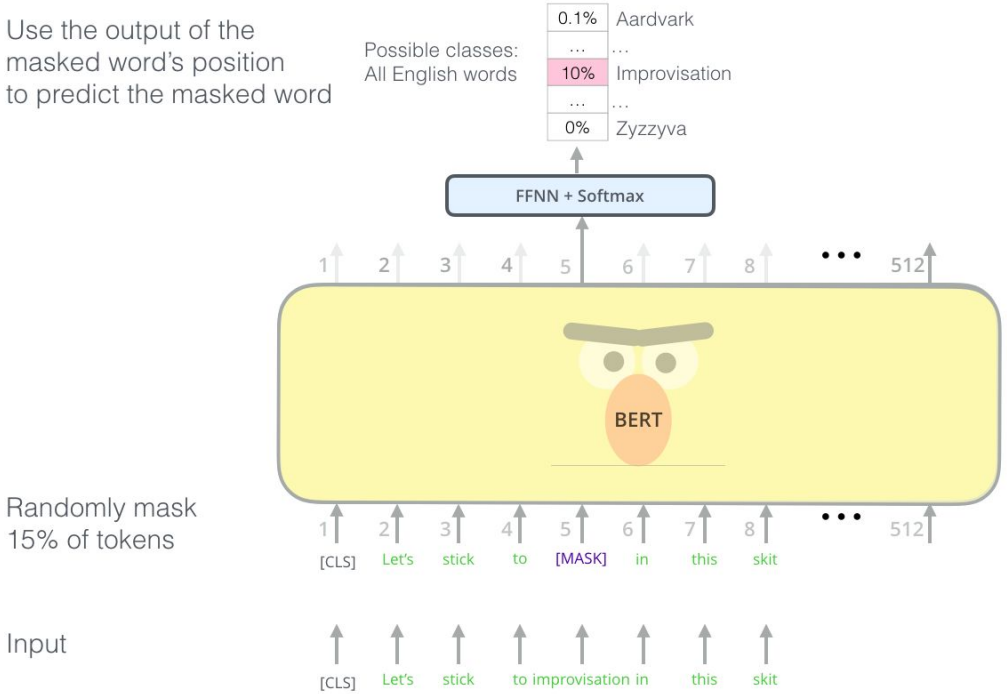
Υλοποιεί ένα μοντέλο βασισμένο σε transformer, του οποίου το γλωσσικό μοντέλο βλέπει προς τα εμπρός και προς τα πίσω(τεχνικά “is conditioned on both left and right context”)

Masked Language Model (MLM)

1. Πριν την τροφοδότηση των ακολουθιών λέξεων στο BERT, 15%των λέξεων που συνθέτουν μια ακολουθία αντικαθίστανται από μία μάσκα [MASK] αποκρύπτοντας έτσι την αρχική τους τιμή.

2. Στη συνέχεια το μοντέλο προσπαθεί να προβλέψει την αρχική τιμή των λέξεων με μάσκα βασισμένο στο περιεχόμενο των υπόλοιπων λέξεων στην ακολουθία που δεν έχουν μάσκα.

BERT: από τους Decoders στους Encoders



Από τεχνικής πλευράς, η πρόβλεψη των όρων [MASK] προϋποθέτει τη προσθήκη ενός επιπέδου ταξινόμησης (classification layer) στην έξοδο του κωδικοποιητή.

Η έξοδος του επιπέδου ταξινόμησης μετασχηματίζεται στις διαστάσεις του λεξιλογίου αφού πολλαπλασιαστεί με τον πίνακα embedding

Εφαρμόζεται η συνάρτηση softmax στο διάνυσμα που έχει παραχθεί για τον υπολογισμό της πιθανότητας κάθε λέξη να αντιπροσωπεύει την αρχική τιμή του όρου [MASK]

BERT: από τους Decoders στους Encoders

Next Sentence Prediction (NSP)

Η επόμενη στρατηγική για την εκπαίδευση του μοντέλου είναι η πρόβλεψη επόμενης ακολουθίας.

- Ουσιαστικά, το μοντέλο τροφοδοτείται με ζευγάρια ακολουθιών και προσπαθεί να προβλέψει αν η δεύτερη ακολουθία βρίσκεται αμέσως μετά την πρώτη στο αρχικό κείμενο.

- Κατά την διαδικασία εκπαίδευσης το 50%των ζευγαριών αποτελείται από διαδοχικές ακολουθίες στο κείμενο, ενώ το υπόλοιπο 50% αποτελείται από ζευγάρια τυχαία επιλεγμένων ακολουθιών.

Και σε αυτή τη στρατηγική ένα επίπεδο ταξινόμησης προστίθεται στην έξοδο του κωδικοποιητή, προσπαθώντας να ταξινομήσει τις εξόδους στις κλάσεις IsNext, NotNext εφαρμόζοντας πάλι τη συνάρτηση softmax.

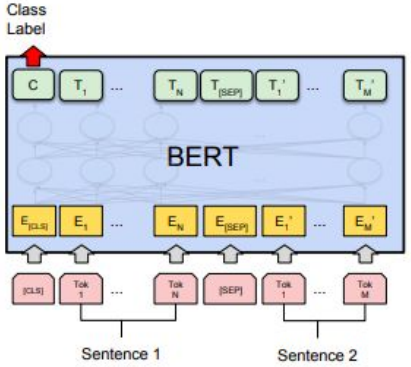
BERT: από τους Decoders στους Encoders

Fine-tuning

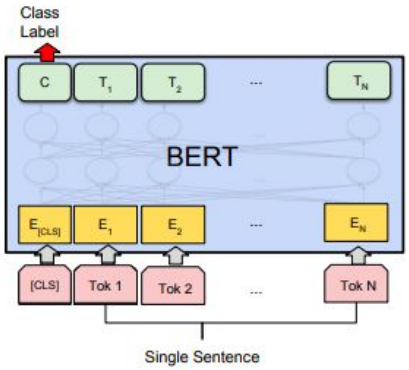
- Τέλος, μόλις ολοκληρωθεί η διαδικασία της προεκπαίδευσης (pretraining) μπορεί να προστεθεί στο τέλος του μοντέλου ένας αποκωδικοποιητής ή ένας ταξινομητής (ρηχό πλήρως συνδεδεμένο δίκτυο) για την προσαρμογή του σε κάποια συγκεκριμένη εφαρμογή.
- Κατά την διαδικασία του fine-tuning μόνο τα επιπρόσθετα επίπεδα ανανεώνουν τις παραμέτρους τους και ίσως μερικά από τα τελευταία επίπεδα του δικτύου, ενώ οι περισσότερες από τις παραμέτρους του συστήματος δεν επηρεάζονται.
- Τα δεδομένα εκπαίδευσης που συνοδεύουν κάθε εφαρμογή θα χρησιμοποιηθούν κατά την διαδικασία fine-tuning για την εκπαίδευση των τελευταίων επιπρόσθετων επιπέδων.

Το BERT παρέχει προεκπαιδευμένα μοντέλα έτοιμα για χρήση, με μοναδική απαίτηση το fine-tuning.

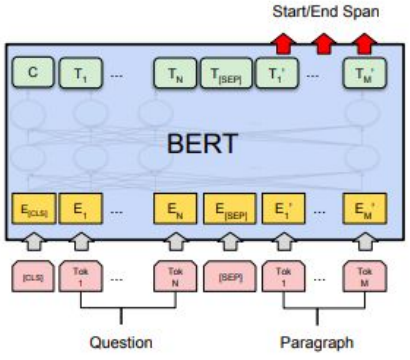
Task specific-Models



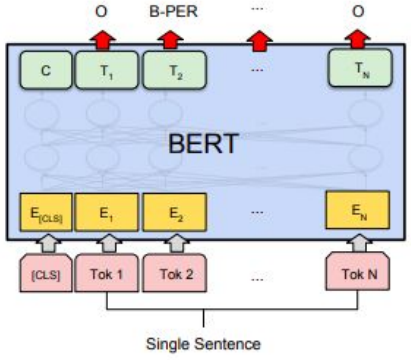
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



(b) Single Sentence Classification Tasks: SST-2, CoLA



(c) Question Answering Tasks: SQuAD v1.1



(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Υλοποιήσεις για NLP

- Sequence-to-Sequence Modeling with nn.Transformer and TorchText
- https://colab.research.google.com/github/tensorflow/tensor2tensor/
- chapter_attention-mechanisms/transformer.ipynb
- http://jalammar.github.io/a-visual-guide-to-using-bert-for-the-first-time/
- NLP Tutorial: Creating Question Answering System using BERT + SQuAD on Colab TPU
- Lit BERT: NLP Transfer Learning In 3 Steps
- BERT — transformers 2.9.0 documentation
- Question Answering with a Fine-Tuned BERT

Υλικό για NLP

- http://web.stanford.edu/class/cs224n
- Attention is all you need; Attentional Neural Network Models | Łukasz Kaiser
- https://ruder.io/a-review-of-the-recent-history-of-nlp
- dipanjanS/hands-on-transfer-learning-with-python · GitHub
- Deep Learning For NLP: Zero To Transformers & BERT
- Neural Transfer Learning for Natural Language Processing, S. Ruder thesis
- https://cs231n.github.io/transfer-learning/#add
- Lecture 7: Training Neural Networks, Part 2
- [1808.01974] A Survey on Deep Transfer Learning
- The Stanford Question Answering Dataset