Στατιστική Μοντελοποίηση και Αναγνώριση Προτύπων
Handwritten Text Recognition Project

Αλέξανδρος Πουπάκης

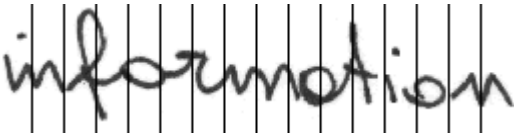
***Περίληψη*—Η αναγνώριση χειρόγραφων κειμένων είναι ένας τομέας εντατικής έρευνας και κατέχει ευρύ επιστημονικό αλλά και εμπορικό ενδιαφέρον. Η παρούσα εργασία προσπαθεί να βελτιώσει το μοντέλο του [1] που είναι μια απλοποιημένη εκδοχή αυτού στο [2]. Το μοντέλο είναι ένα υβριδικό μεταξύ ενός Convolutional Neural Network (CNN), ενός Bidirectional Long Short Term Memory (BLSTM) δικτύου και τέλος ενός Connectionist Temporal Classification (CTC) δικτύου. Ύστερα από ενδελεχείς δοκιμές, μέσα από μια σειρά μεθόδων και τεχνικών, προέκυψε ένα μοντέλο οριακά καλύτερο από αυτό στο [1]. Συμπεραίνουμε, λοιπόν, ότι απαιτείται περαιτέρω μελέτη και εικάζουμε ότι η παρούσα αρχιτεκτονική ίσως είναι κορεσμένη ως προς την επίδοση.**

I. Εισαγωγη

Η αναγνώριση γραπτών κειμένων (Handwritten Text Recognition - HTR) είναι η ικανότητα ενός υπολογιστή, δεχόμενος ως είσοδο χειρόγραφο κείμενο, συνήθως σε μορφή εικόνων, να τις μετατρέπει σε κείμενο. Ο τομέας αυτός είναι επιστημονικά και εμπορικά ενεργός και η βελτίωση ή βελτιστοποίηση της τεχνολογίας αυτής θα είχε σημαντικό αντίκτυπο στη ψηφιοποίηση καθημερινών και ιστορικών κειμένων. Υπάρχει πληθώρα ιστορικών χειρόγραφων κειμένων, των οποίων η συντήρηση είναι αρκετά απαιτητική διαδικασία, αλλά η ψηφιοποίησή τους θα τα καθιστούσε ευανάγνωστα, προσβάσιμα ανά τον κόσμο και αναλλοίωτα στον χρόνο.

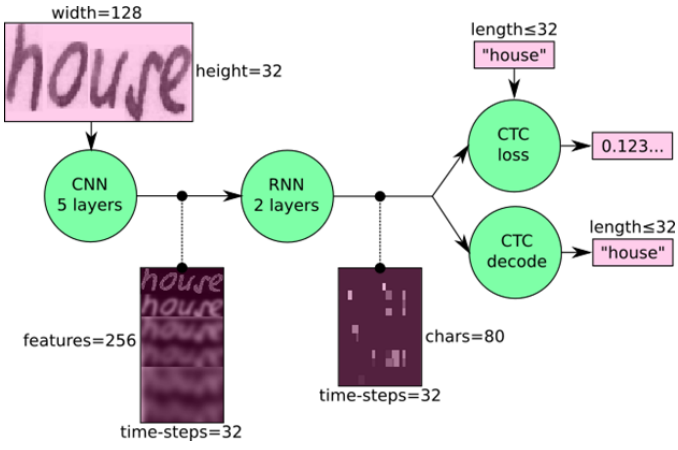
Το μοντέλο στο [1] υλοποιεί μια υβριδική προσέγγιση νευρωνικών δικτύων, συνδυάζοντας τρεις τύπους δικτύων. Σε πρώτο επίπεδο υπάρχει ένα CNN το οποίο εξάγει χαρακτηριστικά από την είσοδο, δηλαδή την εικόνα, για κάθε στήλη της (εφεξής time step). Ο ορισμός του time step γίνεται ξεκάθαρος στο Σχήμα 1. Ένας χαρακτήρας όμως μπορεί να καταλαμβάνει περισσότερα από ένα time steps. Για αυτό, η έξοδος του CNN επεξεργάζεται περαιτέρω από ένα BLSTM δίκτυο το οποίο συνδέει τα χαρακτηριστικά των time steps ώστε να συνθέσει μια έξοδο η οποία λαμβάνει υπόψιν γειτονικά time steps. Τέλος υπάρχει ένα (CTC) δίκτυο μέσω του οποίου υπολογίζεται η «βέλτιστη» – επεξηγείται στην Ενότητα III-A για κάθε μέθοδο – ακολουθία χαρακτήρων. Το CTC παρέχει επίσης το κόστος (loss) κάθε υποσυνόλου των δεδομένων εκπαίδευσης (training batch), το οποίο προσπαθεί να ελαχιστοποιήσει το μοντέλο κατά την εκπαίδευση.

Για το CTC απαιτούνται οι πλήθους time steps είσοδοι για καθένα από τους $n+1$ πιθανούς χαρακτήρες. Οι είσοδοι



Σχήμα 1: Παράδειγμα εικόνας χωρισμένης σε 16 time steps

είναι κάποιας μορφής σκορ για κάθε χαρακτήρα – πχ πιθανότητες. Ο επιπρόσθετος χαρακτήρας είναι το κενό (blank) και δεν πρέπει να συγχέεται με το κενό διάστημα (space), πχ μεταξύ δυο λέξεων. Όταν, με κάποια από τις μεθόδους που παρουσιάζονται στην Ενότητα IV, βρεθεί μια ενδιάμεση ακολουθία εξόδου, μήκους όσο το πλήθος των time steps, πρέπει να γίνει συγχώνευση κάποιων χαρακτήρων ώστε στο τέλος να προκύψει η τελική ακολουθία με το επιθυμητό μήκος. Ο αλγόριθμος πρώτα συγχωνεύει όλους τους διαδοχικούς ίδιους χαρακτήρες και έπειτα αφαιρεί τα κενά. Για παράδειγμα, έστω η ενδιάμεση ακολουθία '-tttooo---'. Αυτή θα μετατραπεί σε '-to-' και τέλος η έξοδος θα είναι "to". Εδώ γίνεται ξεκάθαρη η χρησιμότητα των κενών χαρακτήρων. Αν η ζητούμενη λέξη είναι "too", τότε η ενδιάμεση ακολουθία θα περιείχε τουλάχιστον έναν κενό χαρακτήρα μεταξύ των 'o', πχ '--ttttoo-ooo----' ώστε να μετατραπεί σε '-to-o-' και τελικά να καταλήξουμε στο ζητούμενο.



Σχήμα 2: Το μοντέλο αναφοράς, όπως παρουσιάζεται στο [1]

Το τελικό μοντέλο, όπως φαίνεται στο Σχήμα 2, έχει τη δυνατότητα να αναγνωρίσει συμβολοσειρές μεταβλητού μήκους και αποτελεί εφαλτήριο της παρούσας εργασίας. Οι

μετρικές αξιολόγησης των μοντέλων είναι το Character Error Rate (CER) και το Word Error Rate (WER).

II. Σχετικες Εργασιες - Προσεγγισεις

Έχουν διεξαχθεί πολλές έρευνες στην αναγνώριση χειρόγραφων κειμένων σε ποικίλα πεδία και διάφορες γλώσσες.

Ο J. Puigcerver στο [3] παρουσίασε στοιχεία που αποδεικνύουν ότι τα πολυδιάστατα LSTM (MDLSTM) δεν συνεπάγονται απαραίτητα καλύτερη επίδοση του μοντέλου, στο συγκεκριμένο πρόβλημα. Προκειμένου να το πετύχει αυτό χρησιμοποίησε τα datasets IAM και Rimes. Επίσης, υποστηρίζει ότι το MDLSTM είναι υπολογιστικά ισχυρότερο από τα CNN, όμως η συνεπαγώμενη υπολογιστική επιβάρυνση του συστήματος δεν είναι αναγκαία για την αναγνώριση κειμένου.

Οι B. Moysset και R. Messina στο [4] αξιολόγησαν διάφορες αρχιτεκτονικές νευρωνικών δικτύων που κυμαίνονται από feed-forward έως 1D και 2D LSTMs. Τα αποτελέσματά τους έδειξαν πως τα LSTM δίκτυα είναι αναπόσπαστο κομμάτι των HTR μοντέλων. Καταλήγουν ότι για απλά datasets τα αποτελέσματα δεν διαφέρουν σημαντικά μεταξύ των 1D και 2D LSTM δικτύων, όμως πολύπλοκα datasets όπως το READ επωφελούνται από τα 2D LSTMs. Τέλος, σημειώνουν ότι η εφαρμογή γλωσσικού μοντέλου σε αρχιτεκτονικές που περιέχουν LSTMs δεν βελτιώνει την επίδοση σε ένα μεγάλο πλήθος συνθηκών.

Οι A. Poznanski και L. Wolf στο [5] υλοποίησαν ένα CNN με 12 layers (9 συνελικτικά, 3 πλήρως συνδεδεμένα). Η πρωτότυπη προσέγγισή τους χρησιμοποιεί 19 παράλληλα πλήρως συνδεδεμένα δίκτυα τριών layer για την παραγωγή του attributes vector, αντί για ένα τέτοιο δίκτυο, όπως συνηθίζεται. Χρησιμοποίησαν τα datasets IAM, Rimes και INF/ENIT πετυχαίνοντας state-of-the-art αποτελέσματα. Τέλος, υποστηρίζουν ότι δεν υπάρχει εγγενής δυσκολία στην αναγνώριση χειρόγραφων κειμένων που να αποτρέπει μια μηχανή να επιτύχει ανθρώπινου επιπέδου επίδοση, όπως έχει ήδη γίνει στην αναγνώριση προσώπων.

Τέλος, στο [6], οι A. Chowdhury και L. Vig σχεδίασαν μια υβριδική αρχιτεκτονική με ένα CNN και έπειτα ένα recurrent Encoder-Decoder. Το μοντέλο τους επιτυγχάνει 8.1% CER, 16.7% WER στο IAM και 3.5% CER, 9.6% WER στο Rimes ενώ ταυτόχρονα σημειώνουν αρκετά χαμηλότερες απαιτήσεις σε υπολογισμούς και χρήση μνήμης.

III. Περιγραφη των τεχνικων που ακολουθησαμε

A. CTC Decoding

Από τον τρόπο λειτουργίας του CTC, είναι φανερό ότι για μια δεδομένη λέξη υπάρχουν πολλές πιθανές ενδιάμεσες ακολουθίες. Ιδανικά, με μόνη πληροφορία την πιθανότητα του κάθε συμβόλου σε κάθε time step για μια δεδομένη εικόνα, θα έπρεπε ο αλγόριθμος να υπολογίσει το άθροισμα των πιθανοτήτων των ενδιάμεσων ακολουθιών που καταλήγουν στην ίδια τελική ακολουθία και από το πλήθος των διαφορετικών ακολουθιών να επιλέξει αυτή με τη μέγιστη αθροιστική πιθανότητα. Μια τέτοια εξαντλητική υλοποίηση όμως είναι υπολογιστικά πολύ απαιτητική και είναι εφικτή μόνο για κάποιο (μικρό) πλήθος χαρακτήρων και time steps. Στην παρούσα περίπτωση των 80 χαρακτήρων και 32 time steps, υπάρχουν πάνω από 7.9×10^{60} ενδιάμεσες ακολουθίες, οπότε αναγκαστικά πρέπει να καταφύγουμε σε άλλες τεχνικές.

Υπάρχουν τρεις μέθοδοι για την αποκωδικοποίηση της εισόδου του CTC σε κάποια έξοδο: greedy, beam search και word beam search. Κάθε μια είναι εξέλιξη της προηγούμενης και αντιμετωπίζει προηγούμενα μειονεκτήματα.

Στην **greedy** μέθοδο επιλέγεται ο πιο πιθανός χαρακτήρας (αυτός με το μεγαλύτερο σκορ), μέσα από μια λίστα όλων των δυνατών χαρακτήρων, για κάθε time step και έτσι προκύπτει η τελική ακολουθία-πρόβλεψη του μοντέλου. Ο αλγόριθμος αυτός λέγεται και best path decoding διότι η επιλεγμένη ενδιάμεση ακολουθία έχει τη μεγαλύτερη συνολική πιθανότητα σε σχέση με κάθε άλλη.

Το **beam search decoding** [7] είναι ένας συμβιβασμός μεταξύ απόδοσης και της εξαντλητικής μεθόδου. Ξεκινώντας από μηδέν ενδιάμεσες ακολουθίες, για κάθε time step επεκτείνονται οι υπάρχουσες κατά έναν χαρακτήρα για τους $n+1$ χαρακτήρες και διατηρούνται οι beam width καλύτερες ακολουθίες. Εάν δύο ακολουθίες είναι ισοδύναμες, δηλαδή παράγουν την ίδια τελική ακολουθία, συγχωνεύονται ως μια ενδιάμεση ακολουθία με σκορ το άθροισμα των σκορ των επιμέρους συγχωνευμένων ακολουθιών. Όταν εξαντληθούν τα time steps, επιλέγεται η ακολουθία με το μεγαλύτερο σκορ.

Τέλος, το **word beam search decoding** [8] ενσωματώνει ένα γλωσσικό μοντέλο σε επίπεδο λέξεων. Από τις λέξεις του training set κατασκευάζεται ένα prefix tree και εισάγεται μια κατάσταση για κάθε beam που δείχνει ποιοι χαρακτήρες επιτρέπεται να εισαχθούν αμέσως μετά στο beam. Στην κατάσταση «λέξης» (word state), το beam προσπαθεί να κατασκευάσει μια λέξη οπότε επιτρέπονται μόνο αλφαβητικοί χαρακτήρες, ενώ στην κατάσταση «όχι λέξη» (non word state) επιτρέπονται όλοι οι χαρακτήρες. Η μετάβαση από την κατάσταση non-word στην word επιτρέπεται πάντα, ενώ το αντίστροφο επιτρέπεται μόνο εάν έχει ολοκληρωθεί μια λέξη, δηλαδή εάν ο αλγόριθμος έχει φτάσει σε πιθανό τερματικό κόμβο στο prefix tree. Έτσι, για κάποιο beam, μέσω του prefix tree βρίσκονται οι χαρακτήρες που μπορούν να εισαχθούν στην επόμενη θέση, ώστε τελικά να κατασκευαστεί μια λέξη από αυτό το beam. Με την πληροφορία αυτή, ο αλγόριθμος beam search επεκτείνει κάθε beam με κάθε έναν από τους k επιτρεπτούς – για τη συγκεκριμένη θέση – χαρακτήρες και συνεχίζει κατά τα γνωστά.

Είναι κατανοητό ότι όσο αυξάνεται η περιπλοκότητα των αλγορίθμων αποκωδικοποίησης, τόσο αυξάνεται ο χρόνος εκτέλεσης αλλά και η επίδοση του συστήματος. Για τον λόγο αυτό, ο αλγόριθμος word beam search decoding δεν χρησιμοποιείται στην εκπαίδευση του μοντέλου αλλά μόνο κατά τη χρήση του.

B. Τεχνικές βελτίωσης

Στη προσπάθεια βελτίωσης του μοντέλου, δοκιμάστηκαν οι παρακάτω τεχνικές (ή/και συνδυασμοί αυτών):

1) Τυχαίες μετατοπίσεις της εικόνας εισόδου
2) Εισαγωγή θορύβου στο φόντο
3) Leaky ReLU ως συνάρτηση ενεργοποίησης στο CNN
4) Adam optimizer έναντι του RMSprop
5) Μεγαλύτερες εικόνες εισόδου
6) Δυαδικές εικόνες (άσπρο - μαύρο)
7) Είσοδο στο CNN τον μετασχηματισμό Fourier της αρχικής εικόνας
8) Μείωση των time steps από 32 σε 16
9) 2D έξοδος από το CNN για κάθε time step και επεξεργασία της κάθε 1D εξόδου από ένα ξεχωριστό LSTM
10) 2D έξοδος από το CNN πεπλατυσμένη column-wise σε 1D και επεξεργασία από ένα LSTM
11) Εισαγωγή πλήρως συνδεδεμένου δικτύου μεταξύ του LSTM και του CTC
12) Κανονικοποίηση των εισόδων του CTC
13) Μετατροπή των εισόδων του CTC σε νόμο πιθανότητας (μέσω της softmax)
14) Τροποποίηση της συνάρτησης κόστους ώστε κάθε έξοδος να συνεισφέρει περισσότερο στο κόστος του batch ανάλογα με το Levenshtein distance (edit distance) της ως προς το σωστό label
15) L1, L2, L1L2 regularization και Dropout στο LSTM
16) Δισδιάστατο LSTM
17) Γλωσσικό μοντέλο δεσμευμένων πιθανοτήτων σε επίπεδο συλλαβών (δυάδων γραμμάτων)
18) Εισαγωγή, κατά το validation, της κάθε εικόνας 3 φορές στο μοντέλο με τυχαίες παραμορφώσεις και εισαγωγή στο CTC του μέσου όρου των τανυστών εξόδου του LSTM

Στο [9] φαίνεται ότι η συντριπτική πλειοψηφία των αγγλικών λέξεων (αλλά και άλλων ευρωπαϊκών γλωσσών) δεν ξεπερνούν τους 16 χαρακτήρες, οπότε με τη μέθοδο 8 δεν περιορίζουμε αισθητά το εύρος εφαρμογής του μοντέλου αλλά ίσως βελτιωθεί η απόδοση λόγω του μεγαλύτερου πλάτους ανά time step.

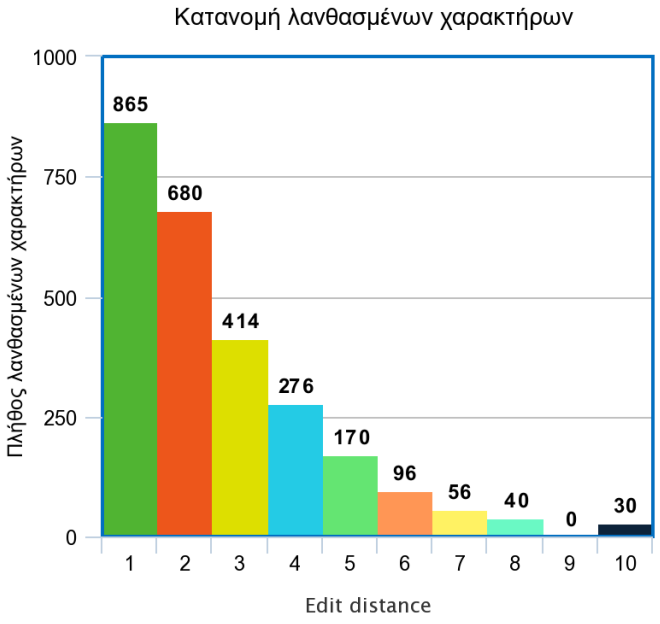
Η μέθοδος 10 προσπαθεί να προσομοιώσει το 2D LSTM και να κρατήσει τυχόν οφέλη του, αποφεύγοντας όμως το επερχόμενο υπολογιστικό και προγραμματιστικό κόστος των 2D LSTM.

Παρατηρώντας την κατανομή του πλήθους των λανθασμένων χαρακτήρων ως προς το edit distance των λανθασμένων labels προκύπτει το ιστόγραμμα του Σχήματος 3. Η μέθοδος 14 στηρίζεται στην ιδέα ότι αν το ιστόγραμμα μετατοπιστεί, έστω και λίγο, προς τα αριστερά, η μείωση του CER θα είναι σημαντική. Η προτεινόμενη συνάρτηση κόστους είναι η

$$\frac{1}{N}\sum_{i=0}^{N} CTC\_loss_i \times (edit\_distance_i + 1)$$

έναντι της συνηθισμένης $\frac{1}{N}\sum_{i=0}^{N} CTC\_loss_i$ όπου CTC_loss_i είναι το σφάλμα του $i-$οστού δείγματος στο batch. Είναι σαφές ότι ο αλγόριθμος CTC υπολογίζει το σφάλμα μέσω της αξιοπιστίας της πρόβλεψης, οπότε έμμεσα εμπεριέχει πληροφορία σχετικά με την εγγυρότητα ή μη της πρόβλεψης. Με την τροποποίηση αυτή όμως, το πλήθος λαθών επηρεάζει άμεσα τη συνάρτηση κόστους, δίνοντας έμφαση στις χειρότερες προβλέψεις.

Στη μέθοδο 17, το γλωσσικό μοντέλο αφορά τον greedy decoder και υλοποιήθηκε σύμφωνα με το κανόνα δεσμευμένης πιθανότητας. Έστω μια ακολουθία διανυσμάτων πιθανοτήτων χαρακτήρων $\{s_0, s_1, ..., s_m\}$ όπου $s_i = [P_i(c_0)\ P_i(c_1)\ ...\ P_i(c_n)]^T$ και $C = \{c_0, c_1, ..., c_n\}$ το σύνολο του αλφαβήτου και έστω ότι τα s_i και $P(c_j|c_k)$ είναι γνωστά. Τότε, μπορεί να κατασκευαστεί η ακολουθία $\{\hat{s}_i\}$ όπου $\hat{s}_i = s_i \odot P(C\ |\ \|s_{i-1}\|_\infty)$. Τα $P(c_j|c_k)$ προκύπτουν από το dataset και η $\{\hat{s}_i\}$ είναι η αρχική $\{s_i\}$, προσαρμοσμένη στο γλωσσικό μοντέλο. Σημειώνεται ότι, στην υλοποίηση αυτή, ισχύει $\hat{s}_0 = s_0$, δηλαδή το διάνυσμα πιθανοτήτων του πρώτου time step παραμένει αναλλοίωτο.



Σχήμα 3: Κατανομή συνολικού πλήθους λανθασμένων χαρακτήρων των λανθασμένων labels ανά edit distance.

Σαφώς, εκτός από τα ανώτερα, δοκιμάστηκαν διάφορα learning rates, batch sizes, μεγέθη kernel, πλήθη features, layers και hidden neurons για το CNN και LSTM αντίστοιχα.

IV. Πειραματικο μερος

Ο κώδικας του μοντέλου αναφοράς στο [1] είχε γραφεί για εκτέλεση σε CPU και από τις πληροφορίες του συντάκτη, χρειάστηκε 18 ώρες εκπαίδευσης. Αυτός ο χρόνος εκτέλεσης είναι απαγορευτικός για πειραματισμούς, οπότε αρχικά εκτελέσθηκε ως είχε στην GPU (απλά εγκαθιστώντας το tensorflow-gpu) και ο συνολικός χρόνος εκτέλεσης ήταν περίπου 6.5 ώρες. Ύστερα από σταδιακές βελτιστοποιήσεις, ο συνολικός χρόνος εκτέλεσης μειώθηκε στα 10-11 λεπτά για το μοντέλο αναφοράς.

Αρχικά εκτελέστηκαν τα baseline πειράματα του Πίνακα I και για την παρούσα χρησιμοποιήθηκε το IAM dataset.

Από τα πειράματα προκύπτει ότι η εκπαίδευση με τον Beam Search Decoder (BS) συνεπάγεται καλύτερη τελική

Model	Validation Error	
	CER	WER
Vanilla[1]		
G	10.62%	73.68%
BS	10.46%	74.00%
WBS	8.07%	84.80%
Vanilla+BS[2]		
G	9.94%	75.26%
BS	9.83%	75.47%
WBS	7.82%	85.37%

Πίνακας I: Baseline πειράματα για κάθε decoder με το μοντέλο του [1].
[1]Εκπαιδευμένο με Greedy Decoder (G).
[2] Εκπαιδευμένο με Beam Search Decoder (BS).

Model	Validation Error	
	CER	WER
G, TS 32, ADAM	11.93%	71.40%
G, RT+RB, TS 32, IS 256x64, LSTM 1	10.93%	75.16%
G, RT+RB, TS 16, IS 256x64, CNN 1, LSTM 2 ("Best G")	**10.40%**	**76.85%**
BS, RT+RB, TS 32	11.53%	72.10%
BS, RT+RB, TS 32, IS 128x64 ("Best BS")	**10.26%**	**74.67%**
BS, RT+RB, TS 32, LReLU	11.38%	71.72%
BS, RT+RB, TS 32, LReLU, IS 128x64, LSTM 1	10.99%	73.11%

Πίνακας II: Πειραματικά αποτελέσματα ενός μέρους των εκπαιδευμένων μοντέλων. RT: Random Translations, RB: Random Background, IS: Image Size, TS: Time Steps.

επίδοση του μοντέλου. Επειδή όμως ο BS αυξάνει τον χρόνο εκτέλεσης των πειραμάτων, έχει εφαρμοστεί σε κάποιες περιπτώσεις, ενώ στις υπόλοιπες τα πειράματα διεξήχθησαν με τον Greedy Decoder (G).

Εκπαιδεύτηκαν συνολικά πάνω από 40 διαφορετικά μοντέλα – τροποποιήσεις του [1] – και τα αποτελέσματα ενός αντιπροσωπευτικού υποσυνόλου παρουσιάζονται στον Πίνακα II. Εξ αυτών, κανένα από τα μοντέλα που εκπαιδεύτηκαν με τον BS δεν ξεπερνά το baseline, ενώ ένα μοντέλο, εκπαιδευμένο με τον G, είναι οριακά καλύτερο. Για τις περιπτώσεις που δεν συμπεριλαμβάνονται στον πίνακα, πχ Fourier, L1/L2 regularization κλπ, τα αποτελέσματα ήταν αρκετά χειρότερα από τα baseline πειράματα και για αυτό παραλείφθηκαν.

Τα LSTM 1 και LSTM 2 αποτελούνταν από 3 layers και 512 και 1024 units αντίστοιχα, σε κάθε LSTM cell, έναντι των 2 layers και 256 units του vanilla. Το CNN 1 είχε (1,1) stride στο τελευταίο layer, έναντι του (1,2) του vanilla.

Στη συνέχεια, στον Πίνακα III φαίνονται τα αποτελέσματα εφαρμογής της τροποποιημένης συνάρτησης κόστους και του γλωσσικού μοντέλου. Οι δυο αυτές μέθοδοι συστηματικά παρήγαγαν χειρότερα αποτελέσματα, ασχέτως του εκάστοτε μοντέλου.

Εκτός από το ποσοστό σφάλματος, έχουν επίσης σημασία ο χρόνος εκπαίδευσης και το μέγεθος των μοντέλων. Στον

Model	Validation Error	
	CER	WER
Vanilla+MLF		
G	11.85%	71.06%
G+LM	12.00%	70.74%

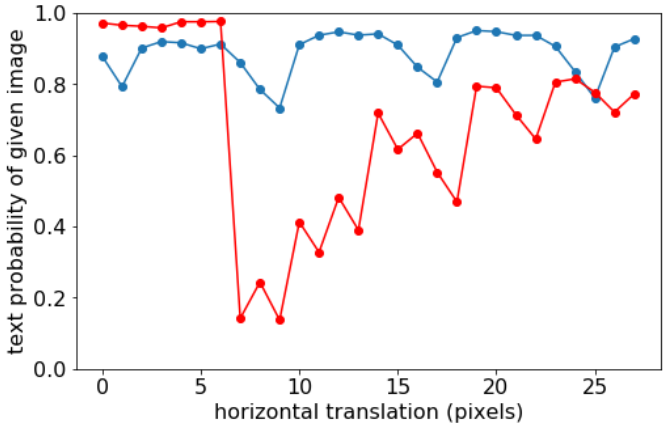
Πίνακας III: Πειράματα με την τροποποιημένη συνάρτηση κόστους (MLF) και το γλωσσικό μοντέλο (LM).

Model	Duration (mins)	Size (MB)	Validation Error	
			CER	WER
Vanilla	10	19.5	10.62%	73.68%
Vanilla+BS	19.5	19.5	9.83%	75.47%
"Best G"	50.5	1057.8	10.40%	76.85%
"Best BS"	33	26.4	10.26%	74.67%

Πίνακας IV: Σύγκριση χρόνου εκπαίδευσης (λεπτά) και μεγέθους των επικρατέστερων μοντέλων (MByte).

Πίνακα IV παρουσιάζονται αυτά τα μεγέθη για τα βέλτιστα μοντέλα σε σύγκριση με αυτά του baseline.

Ένα βασικό στοιχείο των μοντέλων που δεν έχει ακόμα ποσοτικοποιηθεί είναι η ανοχή τους στον θόρυβο και η αμεταβλητότητα του σκορ αξιοπιστίας τις μετατοπίσεις, όπως παρουσιάζεται στο Σχήμα 4. Είναι προφανές ότι τα baseline μοντέλα, επειδή δεν έχουν εκπαιδευτεί με αυτά τα στοιχεία κατά νου, δεν θα παρουσιάζουν τέτοιες ιδιότητες. Η αναφορά όμως πρέπει να γίνει, υπέρ των μοντέλων της παρούσας εργασίας, καθώς αυτές είναι χρήσιμες ως αναγκαίες ιδιότητες σε πραγματικές εφαρμογές. Το vanilla μοντέλο παρουσιάζει ∼25% CER, παρουσία θορύβου και μετατοπίσεων.



Σχήμα 4: Αξιοπιστία πρόβλεψης συναρτήσει οριζόντιας μετατόπισης για δεδομένο δείγμα. Κόκκινο: vanilla, Μπλε: "Best G"

Τέλος, σημειώνεται ότι με την υλοποίηση του δισδιάστατου LSTM του [10] και δεδομένου ότι ο κώδικας του δεν είναι βελτιστοποιημένος, ο χρόνος εκτέλεσης ήταν τεράστιος (∼ 2 ώρες ανά epoch) οπότε δεν έγινε ολοκληρωμένο πείραμα. Όμως, σύμφωνα με το [3], αυτή η μέθοδος δεν είναι αναγκαία για τη βελτίωση του μοντέλου.

V. Συμπερασματα

Απο τα πειράματα γίνεται φανερό ότι το μοντέλο δεν βελτιώθηκε σημαντικά και συστηματικά, παρά το πλήθος και εύρος των τεχνικών που δοκιμάστηκαν. Αυτό ίσως είναι ένδειξη ότι η συγκεκριμένη αρχιτεκτονική και συναφείς τροποποιήσεις της είναι κορεσμένες ως προς την επίδοσή τους στο δοθέν πρόβλημα. Βέβαια, η υλοποίηση του [3] είναι παρόμοια με αυτή του [1] στην οποία βασιστήκαμε, οπότε χρειάζεται περαιτέρω διερεύνηση των λόγων στασιμότητας της επίδοσης στα πειράματά μας καθώς και των δυνατοτήτων αυτής της αρχιτεκτονικής.

Αναφορικά με το γλωσσικό μοντέλο, η υλοποίησή του σε επίπεδο συλλαβών δεν μειώνει το CER, ενώ σε επίπεδο λέξεων, υπό τη μορφή του αλγορίθμου word beam search decoding, παρατηρείται σημαντική μείωσή του. Παρότι το πρώτο έχει εγγενείς περιορισμούς έναντι του δεύτερου, πρέπει να ληφθεί υπόψιν και ο αλγόριθμος για τον οποίο υλοποιήθηκε το πρώτο. Ειδικότερα, η ενσωμάτωση του γλωσσικού μοντέλου συλλαβών στον αλγόριθμο best path decoding ίσως επιφέρει καλύτερα αποτελέσματα σε σχέση με τα αντίστοιχα baseline πειράματα. Να τονιστεί ότι η προσσέγγιση του word beam search decoding δεν αποτελεί πανάκεια καθώς στην περίπτωση όπου κάποιες λέξεις δεν υπάρχουν στο training set αλλά εμφανίζονται στο validation set, ή σε real world εφαρμογές, το μοντέλο θα αδυνατεί να κάνει σωστές προβλέψεις. Η περίπτωση αυτή όμως εύκολα αποφεύγεται.

Για περαιτέρω μελέτη στο συγκεκριμένο πρόβλημα, μια ενδιαφέρουσα εναλλακτική αρχιτεκτονική που αξίζει διερεύνησης είναι αυτή των Quasi-Recurrent Neural Networks (QRNN) του [11], τα οποία εμπλέκουν τη λειτουργία των CNN με recurrent pooling functions.

Τέλος, τα μοντέλα και οι μέθοδοι που παρουσιάστηκαν μπορούν να χρησιμοποιηθούν για αναγνώριση κειμένων άλλων γλωσσών, όσο αυτές έχουν πληθαριθμικά συγκρίσιμο αλφάβητο με τα αγγλικά. Για γλώσσες όπως τα κινέζικα, που έχουν χιλιάδες χαρακτήρες, είναι πιθανό το μοντέλο να μην πετύχαινε τις επιθυμητές επιδόσεις λόγω του περιορισμένου μεγέθους του και του μεγάλου όγκου των κλάσεων.

Αναφορες

[1] H. Scheidl, "Simple HTR", https://github.com/githubharald/SimpleHTR

[2] H. Scheidl, "Handwritten Text Recognition in Historical Documents", https://repositum.tuwien.ac.at/obvutwhs/download/pdf/2874742

[3] J. Puigcerver, "Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition?", *Document Analysis and Recognition (ICDAR), 2017 14th IAPR International Conference*, vol. 1, 67–72. IEEE.

[4] B. Moysset, R. Messina, "Are 2D-LSTM really dead for offline text recognition?", https://arxiv.org/pdf/1811.10899.pdf

[5] A. Poznanski, L. Wolf, "CNN-N-gram for Handwritting Word Recognition", https://www.cs.tau.ac.il/ wolf/papers/CNNNGram.pdf

[6] A. Chowdhury, L. Vig, "An Efficient End-to-End Neural Model for Handwritten Text Recognition", http://bmvc2018.org/contents/papers/0606.pdf

[7] H. Scheidl, "Beam Search Decoding in CTC", https://towardsdatascience.com/beam-search-decoding-in-ctc-trained-neural-networks-5a889a3d85a7

[8] H. Scheidl, "Word Beam Search in CTC", https://towardsdatascience.com/word-beam-search-a-ctc-decoding-algorithm-b051d28f3d2e

[9] V. Marian et al., "CLEARPOND: Cross-Linguistic Easy-Access Resource forPhonological and Orthographic Neighborhood Densities", PLOS ONE, vol. 7, issue 8

[10] P. Peremy, "Multi Dimensional Recurrent Networks", https://github.com/philipperemy/tensorflow-multi-dimensional-lstm

[11] J. Bradbury et al., "Quasi-Recurrent Neural Networks" https://arxiv.org/pdf/1611.01576.pdf