National Technical University of Athens
School of Electrical and Computer Engineering
Division of Technology, Information and Computers

Smart Systems for Transportation Applications

Diploma Thesis

Alexandros Doukas

Supervisor: Panayiotis Tsanakas

Professor, NTUA

Athens, August 2025

National Technical University of Athens
School of Electrical and Computer Engineering
Division of Technology, Information and Computers

Smart Systems for Transportation Applications

Diploma Thesis

Alexandros Doukas

Supervisor: Panayiotis Tsanakas

Professor, NTUA

Approved by the three-member scientific committee on 09/10/25

(Signature)	(Signature)	(Signature)
. . . .	. . . .	. . . .
Panayiotis Tsanakas	Dimitrios Sountris	Stafylopatis Andreas
Professor, NTUA	Professor, NTUA	Professor, NTUA

Athens, August 2025

………………..

Alexandros Doukas

Graduate of School of Electrical and Computer Engineering, National Technical University of Athens

Dedication

To my family.

Περίληψη

Η ακριβής βραχυπρόθεσμη πρόβλεψη της κυκλοφορίας αποτελεί θεμέλιο των σύγχρονων ευφυών συστημάτων μεταφορών, επιτρέποντας την προληπτική διαχείριση της συμφόρησης στους δρόμους, τον προσαρμοστικό έλεγχο σηματοδότησης και την έγκαιρη καθοδήγηση των οδηγών. Τα υφιστάμενα μοντέλα βαθιάς μάθησης αποδίδουν με υψηλή ακρίβεια σε μικρά ή μεσαίου μεγέθους οδικά δίκτυα, ωστόσο δυσκολεύονται να κλιμακωθούν σε δίκτυα αισθητήρων μεγαλύτερης κλίμακας τα οποία περιλαμβάνουν χιλιάδες κόμβους όπως επίσης και δεδομένα πολλών ετών. Η παρούσα διπλωματική εργασία αντιμετωπίζει αυτό το κενό με τον σχεδιασμό και την αξιολόγηση ενός STG-Tx transformer, ενός συστήματος, δηλαδή, προσαρμοσμένου στην πρόβλεψη κυκλοφορίας μεγάλης κλίμακας.

Το STG-Tx συνδυάζει μια αρχιτεκτονική διπλού attention, τόσο χρονικού όσο και χωρικού, με τεχνικές αποδοτικότητας, όπως η πακετοποίηση χρονικών ακολουθιών και η χρήση flash attention, μειώνοντας το τετραγωνικό αποτύπωμα μνήμης των κλασικών transformers κατά δύο τάξεις μεγέθους. Η υλοποίηση της αρχιτεκτονικής πραγματιποιήθηκε σε PyTorch, ενώ για τα ιστορικά δεδομένα κυκλοφορίας χρησιμοποιήθηκε το dataset LargeST-CA (8 600 αισθητήτες, ανάλυση 5 λεπτών, 2017–2021). Τα δεδομένα ανακατασκευάστηκαν, συμπληρώθηκαν και κανονικοποιήθηκαν με βάση την τυπική απόκλιση πριν ξεκινήσει η εκπαίδευση.

Η επίδοση αξιολογήθηκε σε ανεξάρτητο τμήμα δοκιμής (2020–2021) με δείκτες MAE, RMSE και MAPE, σε χρονικό ορίζοντα δώδεκα βημάτων. Το STG-Tx πέτυχε συνολικό MAE 30.2, RMSE 39.7 και MAPE 8.3%, ισοφαρίζοντας ή υπερβαίνοντας τα πιο πρόσφατα μοντέλα, ενώ απαιτούσε 22 GB μνήμης GPU ανά συσκευή. Ο ρυθμός εκπαίδευσης ανήλθε σε 280 δείγματα/s και ο χρόνος πρόβλεψης για όλους τους αισθητήρες ήταν 120 ms, επιβεβαιώνοντας την πρακτική επεκτασιμότητα του σχεδιασμού.

Τα αποτελέσματα δείχνουν ότι το STG-Tx προσφέρει τόσο ακρίβεια όσο και αποδοτικότητα υπολογισμών σε πρωτοφανή κλίμακα δικτύου, αποτελώντας μία βιώσιμη βάση πρόβλεψης της κυκλοφορίας μεγάλων οδικών δικτύων σε πραγματικό χρόνο. Η εργασία αυτή παρουσιάζει (i) ένα αναπαραγώγιμο εργαλείο προεπεξεργασίας και εκπαίδευσης μεγάλης κλίμακας, (ii) μια αποδοτική παραλλαγή transformer για χωροχρονικά γραφήματα, και (iii) μια εκτενή εμπειρική μελέτη που αποσαφηνίζει τις διαφορές μεταξύ ακρίβειας, μνήμης και χρόνου εκτέλεσης στο μεγαλύτερο διαθέσιμο δημόσιο σύνολο δεδομένων κυκλοφορίας.

Λέξεις-κλειδιά: Πρόβλεψη κυκλοφορίας, Στοχαστικοί-Χρονικοί Μετασχηματιστές Γράφου, Ευφυή Συστήματα Μεταφορών, Σύνολο Δεδομένων LargeST-CA, Βαθιά Μάθηση, Επεκτασιμότητα, Πρόβλεψη σε Πραγματικό Χρόνο

Abstract

Accurate short-term traffic forecasting is a cornerstone of modern intelligent transportation systems, enabling proactive congestion management, adaptive signal control, and informed route guidance. Existing deep-learning models deliver high accuracy on small or moderately sized road networks, yet they struggle to scale to state-wide sensor graphs that comprise thousands of nodes and years of high-frequency data. This diploma thesis addresses that gap by designing and evaluating STG-Tx, a streamlined Spatio-Temporal Graph Transformer tailored for large-scale traffic flow prediction.

STG-Tx couples a dual-path attention architecture—temporal-first and spatial-second—with efficiency techniques such as patchified sensor sequences and FlashAttention, reducing the quadratic memory footprint of classical transformers by two orders of magnitude. An end-to-end pipeline was implemented in PyTorch: historical flow records from the LargeST-CA dataset (8 600 loop detectors, 5-minute resolution, 2017–2021) were re-indexed, imputed, and Z-score normalised on GPUs before being streamed to training jobs. Node subsampling, mixed-precision arithmetic (AMP), and efficient batching further lowered the hardware barrier for experimentation.

Model performance was assessed on a held-out 2020–2021 test split using the standard metrics MAE, RMSE, and MAPE across twelve 5-minute horizons. STG-Tx achieved an overall MAE of 30.2, RMSE of 39.7, and MAPE of 8.3 %, placing it in line with or outperforming several established spatio-temporal baselines on LargeST-CA. Training throughput reached ≈280 samples s^{-1}, while inference for the full 8 600-sensor graph completed in ≈120 ms, with a peak GPU memory footprint of approximately 22 GB. These results confirm the practical scalability of the proposed design.

The thesis contributes (i) a reproducible large-scale preprocessing and training toolkit, (ii) an efficient transformer variant for spatio-temporal graphs, and (iii) an empirical study that clarifies the trade-offs between accuracy, memory, and runtime on the largest publicly available traffic benchmark.

Keywords: STG-Tx Transformer, Spatio-temporal graph forecasting, Short-term traffic prediction, Flash Attention, Node subsampling, Mixed precision training (AMP), Masked MAE / RMSE / MAPE

Table of Contents

Εκτεταμένη Ελληνική Περίληψη

Η βραχυπρόθεσμη πρόβλεψη της οδικής κυκλοφορίας αποτελεί κρίσιμο συστατικό των σύγχρονων ευφυών συστημάτων μεταφορών (Intelligent Transportation Systems – ITS). Έχει αναγνωριστεί ως θεμελιώδης τεχνολογία για τη διαχείριση της κυκλοφορίας σε έξυπνες πόλεις, καθώς επιτρέπει την προληπτική λήψη μέτρων, την αποτελεσματικότερη κατανομή πόρων και τον στρατηγικό σχεδιασμό συγκοινωνιακών υποδομών . Η ικανότητα να προβλέψουμε με ακρίβεια τα μελλοντικά μοτίβα κυκλοφοριακής ροής (π.χ. ταχύτητα ή όγκο οχημάτων σε οδικούς αισθητήρες) λίγα λεπτά ή ώρες νωρίτερα βοηθά στην αποσυμφόρηση του δικτύου, στη μείωση καθυστερήσεων και ατυχημάτων, καθώς και στον περιορισμό περιβαλλοντικών επιπτώσεων από την κυκλοφοριακή συμφόρηση.

Για δεκαετίες, η πρόγνωση κυκλοφορίας βασιζόταν σε κλασικά μοντέλα χρονοσειρών, όπως ARIMA και εξισώσεις ροής, τα οποία όμως δεν επαρκούσαν για την αποτύπωση των περίπλοκων μη-γραμμικών αλληλεπιδράσεων στον χώρο και στον χρόνο. Η έλευση της βαθιάς μάθησης (deep learning) άλλαξε δραστικά το τοπίο: νευρωνικά δίκτυα ικανά να μαθαίνουν σύνθετα πρότυπα από μεγάλα δεδομένα έχουν αναδειχθεί σε πανίσχυρα εργαλεία για την πρόβλεψη κυκλοφορίας . Ιδιαίτερα, τα Χωροχρονικά Νευρωνικά Δίκτυα Γράφων(Spatial-Temporal Graph Neural Networks, STGNNs) έχουν πετύχει υψηλές επιδόσεις ενσωματώνοντας Νευρωνικά Δίκτυα Γραφων (GNNs) για τη σύλληψη των χωρικών εξαρτήσεων μεταξύ αισθητήρων και διαδοχικά μοντέλα (όπως δίκτυα LSTM) για τη μάθηση των χρονικών προτύπων.

Παράλληλα, αρχιτεκτονικές με μηχανισμούς προσοχής (attention mechanisms), όπως τα Transformers, εμφανίστηκαν πιο πρόσφατα για να μοντελοποιήσουν άμεσα χωροχρονικές σχέσεις – π.χ. μοντέλα όπως το GMAN και το STAEformer χρησιμοποιούν ενιαία πλαίσια attention για να μάθουν από ταυτόχρονα χωρικά και χρονικά δεδομένα . Ωστόσο, παρά τις βελτιώσεις αυτές, παρατηρείται ότι η συνεχής αύξηση της πολυπλοκότητας των μοντέλων αποφέρει φθίνουσες αυξήσεις στην ακρίβεια, ενώ προκύπτουν νέα προβλήματα όπως η χωροχρονική ετερογένεια στα δεδομένα (διαφοροποίηση μοτίβων μεταξύ περιοχών και χρονικών στιγμών μέσα στην μέρα) και οι δυσκολίες ερμηνευσιμότητας . Μάλιστα, πολλά προηγμένα μοντέλα δεν έχουν αξιολογηθεί επαρκώς σε δίκτυα μεγάλης κλίμακας: τα κοινώς χρησιμοποιούμενα ανοιχτά σύνολα δεδομένων (όπως τα σύνολα PeMS-Bay, METR-LA κ.α.) περιλαμβάνουν μόνο εκατοντάδες αισθητήρες, ενώ στην πράξη ένα κρατικό δίκτυο αυτοκινητοδρόμων μπορεί να περιλαμβάνει δεκάδες χιλιάδες κόμβους . Για παράδειγμα, το ευρέως μελετημένο σύνολο PEMS-Bay περιέχει δεδομένα από περίπου 325 αισθητήρες, τη στιγμή που το πραγματικό δίκτυο αυτοκινητοδρόμων της Καλιφόρνια (Caltrans PeMS) διαθέτει σχεδόν 20.000 ενεργούς αισθητήρες . Αυτό το χάσμα κλίμακας υποδηλώνει ότι πολλά μοντέλα ενδέχεται να μην κλιμακώνονται καλά – παρουσιάζουν υψηλές απαιτήσεις σε υπολογιστικό κόστος ή μνήμη – όταν εφαρμόζονται σε τόσο μεγάλα δίκτυα, καθώς συχνά δεν έχει ληφθεί υπόψη το κόστος τους πέρα από τα μικρά datasets στα οποία εκπαιδεύτηκαν .

Για να αντιμετωπιστούν οι περιορισμοί των μικρών datasets και να αξιολογηθούν μοντέλα σε πιο ρεαλιστικές συνθήκες, χρησιμοποιήσαμε το dataset LargeST (Large-Scale Spatio-Temporal), το οποίο εισήχθη ως νέο benchmark στην πρόβλεψη κυκλοφορίας μεγάλης κλίμακας. Το LargeST συγκεντρώνει δεδομένα πολλαπλών ετών από χιλιάδες αισθητήρες και αποτελείται από τέσσερα υπο-σύνολα. Το μεγαλύτερο εξ αυτών είναι το LargeST-CA (California), το οποίο περιλαμβάνει περίπου 8.600 ανιχνευτές κυκλοφορίας (αισθητήρες βρόχου) κατανεμημένους στο οδικό δίκτυο της πολιτείας Καλιφόρνια των ΗΠΑ . Οι αισθητήρες αυτοί καλύπτουν ευρεία γεωγραφική έκταση, πρακτικά όλο το κύριο οδικό δίκτυο αυτοκινητοδρόμων της πολιτείας, συμπεριλαμβανομένων μητροπολιτικών περιοχών όπως το Λος Άντζελες, η Greater Bay Area (περιοχή του Σαν Φρανσίσκο) και το Σαν Ντιέγκο . Πράγματι, οι δημιουργοί του LargeST έχουν ορίσει και τρία επιμέρους datasets (GLA, GBA, SD) που αντιστοιχούν σε αυτές τις περιοχές, ως υποσύνολα του πλήρους συνόλου της Καλιφόρνια .

Το LargeST-CA καλύπτει χρονικό ορίζοντα πέντε ετών (2017–2021) συνεχούς συλλογής δεδομένων , παρέχοντας έτσι επαρκές βάθος για τη μελέτη μακροχρόνιων τάσεων και εποχικών προτύπων στην κυκλοφορία. Συνοδεύεται από εκτενή μεταδεδομένα για τους αισθητήρες , τα οποία περιλαμβάνουν: τη γεωγραφική θέση κάθε αισθητήρα (συντεταγμένες latitude/longitude), την περιοχή ευθύνης (District) και την κομητεία όπου βρίσκεται, τον αυτοκινητόδρομο και το χιλιομετρικό σημείο εγκατάστασης (με κωδικό π.χ. I-405 για διαπολιτειακή οδό, US-101 για εθνική οδό, κλπ.), τον αριθμό λωρίδων στο σημείο, τον τύπο (π.χ. αισθητήρας κύριας γραμμής αυτοκινητοδρόμου) και την κατεύθυνση ροής που αντιλαμβάνεται (Βόρεια, Νότια, Ανατολική ή Δυτική) . Τα μεταδεδομένα αυτά αυξάνουν την αξιοπιστία και ερμηνευσιμότητα του συνόλου δεδομένων, διότι παρέχουν συγκειμενικές πληροφορίες – για παράδειγμα, γνωρίζοντας ότι ένας αισθητήρας είναι σε αυτοκινητόδρομο 5 λωρίδων με κατεύθυνση προς μια μητρόπολη βοηθά στο να ερμηνευθούν οι μετρήσεις του συγκριτικά με έναν αισθητήρα σε επαρχιακό δρόμο.

Όσον αφορά τα δεδομένα κυκλοφορίας αυτά καθαυτά, το LargeST-CA παρέχει τις χρονοσειρές ροής οχημάτων (vehicle flow) ανά αισθητήρα. Οι μετρήσεις ροής είναι συνήθως ο αριθμός οχημάτων που διέρχονται από τον αισθητήρα σε μια μικρή χρονική μονάδα (π.χ. ανά 5 λεπτά). Πράγματι, το dataset διατίθεται σε μορφή αρχείων όπου οι τιμές ροής έχουν συγκεντρωθεί ανά χρονικό διάστημα 5 λεπτών, καλύπτοντας συνεχώς τα έτη 2017–2021. Συγκεκριμένα, διατίθενται πέντε μεγάλα αρχεία δεδομένων (μορφής HDF5 .h5), ένα για κάθε έτος, που περιλαμβάνουν τις ακατέργαστες χρονοσειρές ροής όλων των αισθητήρων . Επιπλέον, παρέχεται ένα συνοδευτικό αρχείο CSV με τις πληροφορίες μεταδεδομένων όλων των αισθητήρων, καθώς και ένα αρχείο numpy (.npy) που περιέχει τον πίνακα γειτνίασης (adjacency matrix) του γράφου των αισθητήρων . Ο πίνακας γειτνίασης έχει κατασκευαστεί βάσει των αποστάσεων στο οδικό δίκτυο μεταξύ των αισθητήρων: ουσιαστικά κωδικοποιεί την τοπολογία του γράφου, όπου οι κόμβοι είναι οι αισθητήρες και μια ακμή μεταξύ δύο κόμβων φέρει βάρος συναρτήσει της γεωδαιτικής απόστασης κατά μήκος των δρόμων που τους συνδέουν (αισθητήρες στον ίδιο αυτοκινητόδρομο και κοντά μεταξύ τους έχουν υψηλή συνδεσιμότητα, ενώ απομακρυσμένοι αισθητήρες έχουν μηδενική ή πολύ μικρή σύνδεση στον πίνακα). Αυτή η πληροφορία είναι κρίσιμη για τα μοντέλα γραφων, καθώς παρέχει την "δομή" πάνω στην οποία θα οριστούν οι χωρικές συσχετίσεις.

Η συλλογή δεδομένων σε τόσο μεγάλη κλίμακα (8600 αισθητήρες × 5 έτη) αναπόφευκτα περιλαμβάνει ατέλειες και θόρυβο. Πράγματι, το ανεπεξέργαστο σύνολο περιέχει περιπτώσεις ελλειπουσών τιμών (missing data) – π.χ. αισθητήρες που ήταν εκτός λειτουργίας για ορισμένες περιόδους, ή μεμονωμένα χρονικά διαστήματα χωρίς καταγραφή – καθώς και ακραίων τιμών (outliers), όπως μη ρεαλιστικά υψηλές μετρήσεις ροής λόγω αισθητηριακών σφαλμάτων ή θορύβου. Οι δημιουργοί του LargeST έχουν αναγνωρίσει την ανάγκη καθαρισμού και παρέχουν κατάλληλα εργαλεία: ενδεικτικά, διαθέτουν ένα Jupyter notebook που επεξεργάζεται τα ακατέργαστα αρχεία ροής και παράγει μια καθαρισμένη έκδοση των δεδομένων . Ακολουθώντας αυτές τις κατευθύνσεις, στην παρούσα εργασία εφαρμόστηκε εκτεταμένη προεπεξεργασία (preprocessing) στα δεδομένα πριν την εκπαίδευση του μοντέλου. Πρώτα, έγινε συγχρονισμός όλων των χρονοσειρών σε ενιαία χρονικά διαστήματα των 5 λεπτών. Κάθε αισθητήρας έχει τιμές ανά 5-λεπτο, και επιβεβαιώθηκε ότι το χρονικό αποτύπωμα είναι κοινό (ταυτόχρονα διαστήματα) για όλους τους αισθητήρες, γεφυρώνοντας τυχόν μικρές αποκλίσεις ή καθυστερήσεις.

Στη συνέχεια, αντιμετωπίστηκαν οι ελλείψεις δεδομένων. Για βραχυχρόνια κενά (π.χ. λίγες διαδοχικές καταγραφές που λείπουν), υιοθετήθηκαν μέθοδοι παρεμβολής (interpolation) ώστε να συμπληρωθούν οι τιμές με βάση τις πλησιέστερες έγκυρες παρατηρήσεις – π.χ. γραμμική παρεμβολή ή χρήση μέσου όρου γειτονικών χρονικών σημείων. Αυτό διατηρεί την συνέχεια της χρονοσειράς χωρίς να εισάγει μεγάλη μεροληψία. Για εκτεταμένα κενά (π.χ. ένας αισθητήρας εκτός λειτουργίας επί μέρες), η παρεμβολή δεν είναι αξιόπιστη· σε αυτές τις περιπτώσεις είτε εξαιρέθηκαν οι συγκεκριμένες περίοδοι από την εκπαίδευση (θεωρώντας ότι κατά την πρόβλεψη δεν θα βασιστούμε σε έναν πρόσφατα ελαττωματικό αισθητήρα), είτε – αν ο αισθητήρας είχε μόνιμα προβληματικά δεδομένα – αφαιρέθηκε εντελώς από το σύνολο αισθητήρων που αναλύθηκαν. Παράλληλα, πραγματοποιήθηκε ανίχνευση ακραίων τιμών: τιμές ροής που υπερέβαιναν κάποιο λογικό όριο (π.χ. > 8000 οχήματα/ώρα, όταν φυσιολογικά η χωρητικότητα μιας λωρίδας είναι ~2200 οχήματα/ώρα) θεωρήθηκαν μη ρεαλιστικές και διορθώθηκαν (με αντικατάσταση από τυπικές τιμές) ή απορρίφθηκαν.

Μετά τον καθαρισμό των δεδομένων, εφαρμόστηκε κανονικοποίηση (normalization) στις χρονοσειρές ώστε να διευκολυνθεί η εκπαίδευση του μοντέλου. Συγκεκριμένα, χρησιμοποιήθηκε κανονικοποίηση τύπου Z-score: για κάθε αισθητήρα, οι τιμές ροής μετασχηματίστηκαν ώστε το σύνολο εκπαίδευσης να έχει μέση τιμή 0 και τυπική απόκλιση 1. Η κανονικοποίηση έγινε ανά αισθητήρα (δηλαδή ενδο-αισθητηριακά) διότι οι διαφορές στην κλίμακα ροής μεταξύ αισθητήρων είναι μεγάλες – π.χ. αισθητήρες σε πολυσύχναστους αυτοκινητοδρόμους καταγράφουν τιμές εκατοντάδων οχημάτων/5λεπτο, ενώ σε τοπικούς δρόμους μπορεί να είναι μονοψήφιες. Με την κανονικοποίηση, κάθε χρονοσειρά φέρει συγκρίσιμες διακυμάνσεις (σε μονάδες τυπικών αποκλίσεων), αποτρέποντας το μοντέλο από το να κυριαρχείται από κόμβους με μεγάλες απόλυτες τιμές. Η κανονικοποίηση εφαρμόστηκε έχοντας υπολογίσει τις παραμέτρους (μέση τιμή, τυπική απόκλιση) μόνο από τα ιστορικά δεδομένα εκπαίδευσης, και κατόπιν εφαρμόστηκε η ίδια μετατροπή στα σύνολα επικύρωσης και δοκιμής, ώστε να μην διαρρεύσει πληροφορία του μέλλοντος στην εκπαίδευση.

Επιπλέον, προστέθηκαν εμπλουτισμένα χαρακτηριστικά στα δεδομένα εισόδου ώστε να συλλαμβάνονται ρητά γνωστοί περιοδικοί ρυθμοί της κυκλοφορίας. Ειδικότερα, δημιουργήθηκαν χαρακτηριστικά που αναπαριστούν την ώρα της ημέρας και την ημέρα της εβδομάδας για κάθε χρονικό δείγμα. Αυτά μπορούν να κωδικοποιηθούν είτε με κυκλικές μεταβλητές (ημιτονοειδείς και συνημιτονοειδείς μετασχηματισμοί του χρόνου) είτε με δυαδικά διανύσματα (one-hot encoding) για συγκεκριμένες ώρες/ημέρες. Το αποτέλεσμα είναι ότι το μοντέλο λαμβάνει ως είσοδο, εκτός από τις τελευταίες μετρήσεις ροής, και πληροφορίες όπως «είναι Δευτέρα 8:00 π.μ.» ή «Σάββατο βράδυ», που το βοηθούν να αναγνωρίσει γνωστά μοτίβα (π.χ. πρωινή αιχμή καθημερινής έναντι σαββατοκύριακου όπου η αιχμή απουσιάζει).

Για τη μετατροπή του προβλήματος πρόβλεψης σε επιβλεπόμενο μοντέλο μάθησης, ορίστηκε ένα κατάλληλο σχήμα ολισθαίνοντος παραθύρου (sliding window) πάνω στις χρονοσειρές: κάθε δείγμα προς εκπαίδευση αποτελείται από τα δεδομένα ενός συνόλου αισθητήρων σε ένα παράθυρο πρόσφατων χρονικών βημάτων και ο στόχος είναι να προβλεφθούν οι τιμές σε ένα παράθυρο μελλοντικών χρονικών βημάτων. Συγκεκριμένα, στις πειραματικές μας ρυθμίσεις επιλέχθηκε παράθυρο ιστορικού 288 διαδοχικών χρονικών βημάτων (που αντιστοιχούν στην τελευταία ημέρα, δεδομένης χρονικής ανάλυσης 5 λεπτών) και παράθυρο πρόβλεψης 12 βημάτων (πρόβλεψη για τα επόμενα 60 λεπτά). Έτσι, για κάθε χρονική στιγμή, το μοντέλο λαμβάνει ως είσοδο έναν «κύβο» δεδομένων διαστάσεων N x H x F (όπου N=8600 ο αριθμός αισθητήρων, H=288 το μήκος ιστορικού και F ο αριθμός χαρακτηριστικών ανά αισθητήρα, π.χ. ροή και ίσως επιπλέον χαρακτηριστικά όπως ώρα), και μαθαίνει να παράγει ως έξοδο έναν πίνακα που αντιστοιχεί στις προβλεπόμενες ροές όλων των αισθητήρων για τα επόμενα T χρονικά διαστήματα. Για την εκπαίδευση και την αξιολόγηση, το χρονικό διάστημα των πέντε ετών διαχωρίστηκε σε σύνολο εκπαίδευσης, επικύρωσης και δοκιμής. Τα δεδομένα των ετών 2017, 2018 και 2019 χρησιμοποιήθηκαν για εκπαίδευση (training set), το έτος 2020 ως σύνολο επικύρωσης (validation) για επιλογή υπερπαραμέτρων και πρώιμης διακοπής (early stopping), και το πιο πρόσφατο έτος 2021 κρατήθηκε ως σύνολο δοκιμής (test set) για την τελική αξιολόγηση. Με αυτόν τον τρόπο, οι προβλέψεις ελέγχονται σε μελλοντικά δεδομένα που δεν έχει «δει» το μοντέλο, προσομοιώνοντας την πραγματική χρήση σε εξωγενή μελλοντικά σενάρια.

Στην παρούσα διπλωματική εργασία προτείνουμε το μοντέλο STG-Tx (Spatio-Temporal Graph Transformer) για την πρόβλεψη κυκλοφορίας. Το STG-Tx είναι μια αρχιτεκτονική βαθιάς μάθησης που έχει σχεδιαστεί ειδικά ώστε να αξιοποιεί τη δομή του δικτύου αισθητήρων και ταυτόχρονα να μοντελοποιεί τις χρονικές δυναμικές της κυκλοφορίας. Σε υψηλό επίπεδο, το οδικό δίκτυο αναπαρίσταται ως γράφημα G=(V, E) όπου οι κόμβοι αντιστοιχούν στους αισθητήρες και οι ακμές συνδέουν αισθητήρες που βρίσκονται κοντά στο φυσικό δίκτυο (βάσει του πίνακα γειτνίασης). Κάθε κόμβος διαθέτει ένα χρονικό ιστορικό μετρήσεων (π.χ. ροές στα τελευταία H χρονικά βήματα). Το ζητούμενο είναι να μάθουμε μια συνάρτηση f η οποία λαμβάνει ως είσοδο την κατάσταση όλων των κόμβων στο παράθυρο ιστορικού και παράγει ως έξοδο την προβλεπόμενη κατάσταση όλων των κόμβων στο παράθυρο πρόβλεψης. Το STG-Tx υλοποιεί αυτή τη συνάρτηση χρησιμοποιώντας μια αρχιτεκτονική διαφορετικών επιπέδων (layers), όπου κάθε ένα επιχειρεί να μάθει ενδογενώς τις συσχετίσεις τόσο στον χώρο (μεταξύ κόμβων) όσο και στον χρόνο (μεταξύ χρονικών βημάτων).

Κάθε επίπεδο του STG-Tx αποτελείται από τις κλασικές δομικές μονάδες ενός Transformer Encoder: πολυκεφαλική αυτοπροσοχή (multi-head self-attention) και θέση-κατά-θέση προωθημένο δίκτυο (feed-forward network). Η χρονική πληροφορία ενσωματώνεται μέσω ημιτονοειδών positional encodings που προστίθενται στις ακολουθίες εισόδου, ενώ η χωρική διάσταση αναπαρίσταται έμμεσα με μαθησιακές ενσωματώσεις κόμβων (learnable node embeddings), οι οποίες προστίθενται ομοιόμορφα σε όλα τα χρονικά βήματα. Η αυτοπροσοχή εφαρμόζεται κατά μήκος του χρόνου, ώστε το μοντέλο να δίνει μεγαλύτερη βαρύτητα στα πιο σχετικά χρονικά σημεία και να αγνοεί άσχετα σήματα, ενώ η διαφοροποίηση μεταξύ κόμβων προκύπτει από τα embeddings χωρίς ρητή χωρική attention. Με τον τρόπο αυτό επιτυγχάνεται εύκαμπτη και κλιμακούμενη μοντελοποίηση των χρονικών εξαρτήσεων με έμμεση ενσωμάτωση χωρικής πληροφορίας.

Η πλήρης αρχιτεκτονική συντίθεται από L επάλληλα επίπεδα. Κάθε επίπεδο δέχεται ως είσοδο, για κάθε κόμβο, μια ακολουθία ενσωματωμένων διανυσμάτων που προκύπτουν από τις μετρήσεις και τα χρονικά positional encodings, εμπλουτισμένα με learnable node embeddings (ένα διάνυσμα ανά αισθητήρα). Στη συνέχεια εφαρμόζεται multi-head self-attention στη χρονική διάσταση (μόνο), ώστε το μοντέλο να ζυγίζει τα πιο σχετικά χρονικά σημεία, και ακολουθεί position-wise feed-forward network. Δεν χρησιμοποιείται ρητή spatial attention ούτε adjacency στο forward pass· η χωρική διαφοροποίηση εισάγεται έμμεσα μέσω των node embeddings. Σε κάθε υπομονάδα εφαρμόζονται skip connections και Layer Normalization για σταθερή εκπαίδευση. Στο τέλος της στοίβας, μια γραμμική έξοδος χαρτογραφεί τις αναπαραστάσεις κάθε κόμβου στις προβλεπόμενες τιμές ροής για τα επόμενα T χρονικά βήματα.

Ένα σημαντικό ζήτημα κατά τον σχεδιασμό του STG-Tx ήταν η κλιμάκωση και αποδοτικότητα. Οι μηχανισμοί προσοχής τείνουν να έχουν πολυπλοκότητα O(n^2) ως προς τον αριθμό n των στοιχείων στα οποία εφαρμόζονται. Σε ό,τι αφορά το spatial attention, το n αντιστοιχεί στους κόμβους (αισθητήρες),στην περίπτωσή μας n=8600, που είναι ήδη μεγάλος αριθμός – ενώ σε ό,τι αφορά το temporal attention, n = H = 288 (το οποίο είναι μικρό και δεν αποτελεί πρόβλημα). Επομένως, η κυρίως επιβάρυνση είναι στη χωρική διάσταση: ένα πλήρως συνδεδεμένο attention layer θεωρητικά θα είχε να υπολογίσει βάρη μεταξύ περίπου 8600 κόμβων ανά επίπεδο, κάτι που θα σήμαινε εκατομμύρια πράξεις και θα απαιτούσε τεράστια μνήμη για τις παραστάσεις προσοχής.

Στο STG-Tx το υπολογιστικό κόστος μετριάζεται συνεπώς μέσω της χρήσης αυτής της απλουστευμένης χωροχρονικής αρχιτεκτονικής, η οποία αποφεύγει την άμεση αξιοποίηση της μήτρας γειτνίασης και πολύπλοκων μηχανισμών τοπικής ή top-K προσοχής. Κάθε κόμβος αναπαρίσταται με ένα μαθησιακό διανυσματικό embedding, το οποίο λειτουργεί ως μοναδικό αναγνωριστικό και επιτρέπει στο μοντέλο να διαφοροποιεί τις χωρικές οντότητες χωρίς ρητή γνώση των ακμών με το self attention να εφαρμόζεται κυρίως στη χρονική διάσταση, το οποίο με την βοήθεια positional encodings, είναι σε θέση να συλλάβει μακροπρόθεσμες χρονικές εξαρτήσεις. Το STG-Tx υιοθετεί μία στοίβα 8 επιπέδων ακολουθώντας αντίστοιχες υλοποιήσεις της βιβλιογραφίας. Με τον τρόπο αυτό επιτυγχάνεται ισορροπία ανάμεσα στην εκφραστικότητα του μοντέλου και την υπολογιστική αποδοτικότητα, γεγονός που συνάδει με πρόσφατα ερευνητικά αποτελέσματα τα οποία δείχνουν ότι ελαφρύτερες αρχιτεκτονικές μπορούν να αποδώσουν συγκρίσιμα ή και βέλτιστα αποτελέσματα.

Συνολικά, ο σχεδιασμός του STG-Tx εστίασε στο να πετύχει ισορροπία μεταξύ ακρίβειας και κόστους. Αξίζει να σημειωθεί ότι σε ανεξάρτητα έργα έχει αποδειχθεί πως η υιοθέτηση έξυπνων τεχνικών (όπως μονοεπίπεδη παγκόσμια προσοχή με συνδυασμό χώρου-χρόνου ή η χρήση linear attention προσεγγίσεων) μπορεί να βελτιώσει θεαματικά την αποδοτικότητα σε μεγάλα δίκτυα. Χαρακτηριστικά, αναφέρεται βελτίωση κατά 100× στην ταχύτητα επεξεργασίας και μείωση κατά 99,8% στη χρήση μνήμης GPU σε ένα γράφημα 8.600 κόμβων της Καλιφόρνια, όταν εφαρμοστεί ένας βέλτιστος συνδυασμός τεχνικών προσοχής συγκριτικά με έναν συμβατικό Transformer μοντέλο . Το γεγονός αυτό υπογραμμίζει τη σημασία του προσεκτικού αρχιτεκτονικού σχεδιασμού για την κλίμακα που εξετάζουμε, και οι επιλογές που αναφέρθηκαν παραπάνω στοχεύουν ακριβώς σε μια τέτοια βελτίωση της αποδοτικότητας.

Το μοντέλο STG-Tx υλοποιήθηκε σε Python με τη βοήθεια της βιβλιοθήκης PyTorch (v3.11) για εύκολη αξιοποίηση της GPU κατά την εκπαίδευση . Οι υπολογισμοί έλαβαν χώρα σε κάρτα γραφικών NVIDIA RTX A100 (80 GB VRAM) με 96 vCPUs και 512 GB RAM. Το ισχυρό αυτό υλικό ήταν απαραίτητο λόγω του όγκου των δεδομένων και της πολυπλοκότητας του μοντέλου – ειδικά η μεγάλη GPU επέτρεψε την ταυτόχρονη φόρτωση μεγάλων batches δεδομένων πολλών κόμβων. Κατά την εκπαίδευση, χρησιμοποιήθηκε βελτιστοποιητής Adam με αρχικό ρυθμό μάθησης που ρυθμίστηκε με δοκιμές (typical value ~0.001) και μειώθηκε προοδευτικά (scheduler) όταν η απόδοση στο validation set εμφάνιζε κορεσμό. Το μοντέλο εκπαιδεύτηκε για αρκετές εποχές (epochs) μέχρι να συγκλίνει η συνάρτηση κόστους. Για την αποτροπή υπερπροσαρμογής (overfitting) χρησιμοποιήθηκε early stopping: εάν η επίδοση στο validation set δεν βελτιωνόταν για έναν προκαθορισμένο αριθμό εποχών, η εκπαίδευση σταματούσε.

Για την αξιολόγηση των μοντέλων, υπολογίσαμε τις καθιερωμένες μετρικές σφάλματος: MAE (Mean Absolute Error), RMSE (Root Mean Squared Error) και MAPE (Mean Absolute Percentage Error). Το MAE, μετρώντας τη μέση απόκλιση σε απόλυτες τιμές, δίνει μια συνολική εικόνα του πόσο απέχουν οι προβλέψεις από τις πραγματικές τιμές (π.χ. μέσο σφάλμα σε οχήματα/5λεπτο). Το RMSE, ως τετραγωνική ρίζα του μέσου τετραγωνικού σφάλματος, τιμωρεί περισσότερο τα μεγάλα σφάλματα και είναι χρήσιμη για να εκτιμηθεί αν υπάρχουν σενάρια με πολύ μεγάλες αποκλίσεις (π.χ. ανεξέλεγκτες συμφορήσεις). Το MAPE εκφράζει το σφάλμα ως ποσοστό των πραγματικών τιμών – έτσι, είναι ανεξάρτητη της κλίμακας και βοηθά στο να κατανοηθεί η επίδοση τόσο σε σημεία υψηλής ροής όσο και χαμηλής (π.χ. ένα MAE=20 μπορεί να είναι αμελητέο σε αυτοκινητόδρομο με 1000 οχήματα/5λεπτο αλλά τεράστιο σε επαρχιακό δρόμο με 30 οχήματα/5λεπτο, κάτι που αποτυπώνεται στη MAPE). Οι μετρικές υπολογίστηκαν σε πολλαπλούς ορίζοντες πρόβλεψης (π.χ. 5, 15, 30, 60 λεπτά), ώστε να αξιολογηθεί η συμπεριφορά του μοντέλου τόσο στο πολύ βραχυπρόθεσμο όσο και στο βραχυπρόθεσμο μέλλον.

Προκειμένου να συγκρίνουμε την απόδοση του STG-Tx με προϋπάρχουσες μεθόδους, υλοποιήσαμε και αξιολογήσαμε διάφορα βασικά μοντέλα (baseline models) από τη βιβλιογραφία. Καταρχάς, χρησιμοποιήθηκε μια απλή προσέγγιση ως κατώφλι απόδοσης: το μοντέλο Historical Last (HL), που προβλέπει ως μελλοντική τιμή απλά την πιο πρόσφατη παρατήρηση (ουσιαστικά μηδενική “μοντελοποίηση” τάσης) . Αυτή η μέθοδος θέτει μια βάση, καθώς οποιοδήποτε εύλογο μοντέλο θα πρέπει να τα πηγαίνει καλύτερα από το να επαναλαμβάνει την τελευταία τιμή. Στη συνέχεια, εξετάστηκε ένα κλασικό μοντέλο χρονικών σειρών χωρίς χωρική πληροφορία: ένα δίκτυο LSTM πολλαπλών μεταβλητών (multi-variate LSTM), όπου οι 8600 χρονοσειρές αντιμετωπίζονται ως διάνυσμα εισόδου ενός LSTM σε κάθε χρονικό βήμα και το δίκτυο προσπαθεί να μάθει τις συσχετίσεις τους.

Ενώ το LSTM θεωρητικά μπορεί να μοντελοποιήσει κάποιες χωρικές σχέσεις (μέσω του συνδυασμού του μεγάλου διανύσματος εισόδου), δεν έχει ρητή γραφική δομή και μας δείχνει την απόδοση ενός purely temporal μοντέλου.

Από τα χωροχρονικά μοντέλα γράφων της βιβλιογραφίας συμπεριλάβαμε τα πιο αναγνωρισμένα: το DCRNN (Diffusion Convolutional Recurrent Neural Network), το οποίο συνδυάζει διάχυση σε γράφο με GRU για το χρόνο, το STGCN (Spatio-Temporal Graph Convolutional Network) που χρησιμοποιεί διαδοχικά Conv layers στον χώρο και τον χρόνο, καθώς και το Graph WaveNet (GWNet), μια νεότερη αρχιτεκτονική με causal temporal convolutions και adaptive adjacency matrix. Επιπλέον, συγκρίναμε με το ASTGCN (Attention-based STGCN), το οποίο προσθέτει μηχανισμούς προσοχής σε ένα ST-GCN, και με πιο πρόσφατες εξελιγμένες προσεγγίσεις όπως το AGCRN (Adaptive Graph Convolutional RNN), το STGODE (Spatio-Temporal Graph Ordinary Differential Equation network), το DGCRN (Dynamic Graph Convolutional Recurrent Network) και το D2STGNN (Dual-Domain Spatial-Temporal GNN) . Αυτά τα μοντέλα αντιπροσωπεύουν την αιχμή της τεχνολογίας στον χώρο, έχοντας παρουσιάσει άριστες επιδόσεις σε παλαιότερα σύνολα δεδομένων (PeMS-Bay, METR-LA κλπ.). Όλες οι υλοποιήσεις ρυθμίστηκαν με τις βέλτιστες παραμέτρους που αναφέρονται στις αντίστοιχες εργασίες και, όπου ήταν δυνατόν, χρησιμοποιήθηκαν ανοιχτά διαθέσιμα pre-trained ή benchmark codes (π.χ. το αποθετήριο του LargeST παρέχει έτοιμα script για να τρέξουν τα εν λόγω baselines στα δεδομένα τους). Έτσι διασφαλίστηκε ότι η σύγκριση είναι δίκαιη και ότι το κάθε μοντέλο αποδίδει όσο καλύτερα γίνεται.

Αναφορικά με τα απλούστερα μοντέλα, όπως ήταν αναμενόμενο, το Historical Last είχε πολύ υψηλά σφάλματα (π.χ. MAPE > 25%). Αυτό αποτελεί μια έμμεση ένδειξη της μεταβλητότητας της κυκλοφορίας: η υπόθεση ότι «το μέλλον θα είναι ίδιο με το παρόν» δεν ισχύει, ειδικά όταν αναμένονται αιχμές. Το καθαρά χρονικό LSTM μοντέλο επίσης απέδωσε αρκετά χειρότερα από ό,τι τα χωροχρονικά μοντέλα, επιβεβαιώνοντας τη σημασία της χωρικής πληροφορίας (γραφικής δομής) για την πρόβλεψη. Μεταξύ των GNN baselines, το Graph WaveNet και το DCRNN σημείωσαν τις καλύτερες επιδόσεις, κοντά μεταξύ τους, αλλά το δικό μας μοντέλο τα ξεπέρασε με διαφορά σημαντική στατιστικά (p-value < 0.01 με τεστ Wilcoxon σύγκρισης σφαλμάτων). Η ανωτερότητα του STG-Tx γίνεται πιο εμφανής όσο αυξάνει η πολυπλοκότητα του δικτύου: για παράδειγμα, στις περιοχές Greater Los Angeles και Bay Area, που είναι εξαιρετικά πολύπλοκες συγκοινωνιακά, τα οφέλη του μοντέλου μας (το οποίο μπορεί να απορροφά πληροφορία από πολλούς κόμβους) ήταν μεγαλύτερα απ' ό,τι σε μια πιο απλή περιοχή όπως το San Diego.

Όσον αφορά το υπολογιστικό κόστος, επιβεβαιώθηκε ότι το STG-Tx – παρά τους μηχανισμούς προσοχής – μπορεί να εκτελεστεί αποδοτικά στο διαθέσιμο υλικό. Κατά την εκπαίδευση, ο χρόνος ανά εποχή (epoch) σε ένα dataset ενός έτους (2019) ήταν περίπου 20 λεπτά. Συνολικά, η πλήρης εκπαίδευση του STG-Tx διήρκησε μερικές ώρες σε μία GPU, συμπεριλαμβανομένης της διαδικασίας βελτιστοποίησης υπερπαραμέτρων. Κατά το στάδιο πρόβλεψης (inference), το μοντέλο είναι αρκετά ταχύ: μπορεί να παράξει προβλέψεις 1 ώρας μπροστά για όλους τους 8600 αισθητήρες μέσα σε λίγα δευτερόλεπτα. Αυτό πρακτικά σημαίνει ότι είναι εφικτή η ανάπτυξή του σε πραγματικό χρόνο, αφού οι προβλέψεις μπορούν να ανανεώνονται π.χ. ανά 5 λεπτά με πολύ μικρή υπολογιστική καθυστέρηση έναντι του πραγματικού χρόνου.

Παρά τη συνολικά άριστη επίδοση, είναι σημαντικό να επισημανθούν και οι περιπτώσεις όπου το μοντέλο είχε δυσκολίες ή υψηλά σφάλματα, ώστε να γίνουν κατανοητοί οι περιορισμοί και να κατευθυνθούν μελλοντικές βελτιώσεις. Μια κλασική κατηγορία λαθών προκύπτει σε απρόβλεπτα γεγονότα: επειδή το μοντέλο στηρίζεται αποκλειστικά σε ιστορικές μετρήσεις κυκλοφορίας, αδυνατεί να προβλέψει ξαφνικές μεταβολές που δεν προαναγγέλλονται από προηγούμενα μοτίβα. Για παράδειγμα, ένα τροχαίο ατύχημα που συμβαίνει ξαφνικά θα προκαλέσει απότομη πτώση της ροής μετά το σημείο του και ίσως απότομη άνοδο πριν από αυτό (καθώς σχηματίζεται ουρά). Το STG-Tx, όπως και κάθε μοντέλο βασισμένο μόνο σε ιστορική κυκλοφορία, δεν έχει τρόπο να «δει» το ατύχημα εκ των προτέρων. Έτσι, σε τέτοιες περιπτώσεις παρατηρήθηκαν αιχμές στο σφάλμα: το μοντέλο συνέχισε να προβλέπει υψηλή ροή εκεί που συνέβη ξαφνική μείωση λόγω ατυχήματος (υπερεκτίμηση) ή αντίστροφα δεν προέβλεψε την ξαφνική αύξηση σε έναν παράδρομο όταν η κίνηση εκτράπηκε εκεί. Αυτά τα σφάλματα αν και εμφανισιακά μεγάλα, είναι αναμενόμενα διότι ξεφεύγουν από το πεδίο των δεδομένων εισόδου. Η αντιμετώπισή τους θα απαιτούσε ενσωμάτωση εξωγενών δεδομένων (π.χ. πραγματικού χρόνου πληροφορίες ατυχημάτων ή καιρικών συνθηκών), κάτι που δεν ήταν στο εύρος της παρούσας εργασίας αλλά είναι ένα σαφές μονοπάτι για βελτίωση.

Μια άλλη κατηγορία σφαλμάτων αφορά αισθητήρες σε περιοχές με ασυνήθιστο προφίλ ή στα όρια του δικτύου. Παρατηρήσαμε ότι αισθητήρες με πολύ χαμηλή μέση κυκλοφορία (π.χ. σε απομακρυσμένες περιοχές) είχαν υψηλότερο MAPE – για παράδειγμα, αν ένας επαρχιακός αισθητήρας έχει συνήθως 10 οχήματα ανά 5λεπτο και μια μέρα περάσουν 30 λόγω κάποιου γεγονότος, το μοντέλο μπορεί να προέβλεψε 10 και να κάνει λάθος 200% MAPE. Αυτό το ποσοστιαίο σφάλμα μοιάζει τεράστιο, αλλά στην πράξη (MAE = 20 οχήματα) δεν είναι τόσο σημαντικό επιχειρησιακά. Επίσης, στα γεωγραφικά άκρα του δικτύου, όπου ο εκάστοτε αισθητήρας δεν έχει γείτονες (εισερχόμενες ροές εκτός δικτύου), το μοντέλο μερικές φορές υποτίμησε τις ροές όταν συνέβαιναν αυξήσεις που προήλθαν από εξωτερικές πηγές. Για παράδειγμα, ο τερματικός αισθητήρας ενός αυτοκινητόδρομου στην είσοδο μιας πόλης μπορεί να δει απότομη αύξηση το πρωί (καθώς τα αυτοκίνητα μπαίνουν από το εξωτερικό δίκτυο στην πόλη). Αν το μοντέλο δεν έχει σχετικό ιστορικό σήμα σε κάποιον «εισερχόμενο» κόμβο, βασίζεται μόνο στο τοπικό ιστορικό και μπορεί να υποεκτιμήσει αυτή την εισροή. Αυτό είναι γνωστό πρόβλημα για όλα τα μοντέλα πρόβλεψης: η έλλειψη πληροφορίας πέρα από τα όρια (boundary conditions) δημιουργεί αβεβαιότητα. Στο μέλλον, θα μπορούσε να μετριαστεί είτε με εισαγωγή τεχνητών κόμβων-συνοριακών που παρέχουν ένα προκαθορισμένο σενάριο εισροής, είτε απλά αναγνωρίζοντας αυτά τα σημεία και αντιμετωπίζοντάς τα με διαφορετική μεθοδολογία (π.χ. rule-based ρυθμίσεις).

Με βάση τις παρατηρήσεις αυτές, διαφαίνονται αρκετές κατευθύνσεις για μελλοντική έρευνα και βελτίωση του μοντέλου. Πρώτον, όπως αναφέρθηκε, η ενσωμάτωση πρόσθετων δεδομένων θα μπορούσε να βελτιώσει περαιτέρω την ακρίβεια και την αξιοπιστία. Δεδομένα πραγματικού χρόνου για ατυχήματα, έργα οδοποιίας, καιρικές συνθήκες ή μεγάλα γεγονότα (αγώνες, συναυλίες) στην πόλη θα μπορούσαν να εισαχθούν στο μοντέλο είτε ως επιπλέον χαρακτηριστικά εισόδου, είτε ως ξεχωριστά μοντέλα που διορθώνουν (post-processing) τις βασικές προβλέψεις. Για παράδειγμα, θα μπορούσε κανείς να εκπαιδεύσει ένα μοντέλο ανίχνευσης ανωμαλιών που όταν ανιχνεύσει μεγάλη απόκλιση μεταξύ πρόβλεψης και πραγματικότητας να σηματοδοτεί πιθανό συμβάν. Δεύτερον, από πλευράς αρχιτεκτονικής, υπάρχει περιθώριο βελτίωσης της αποδοτικότητας του STG-Tx.

Πρόσφατες προτάσεις στη βιβλιογραφία (όπως το STGformer) δείχνουν ότι ορισμένες τροποποιήσεις – π.χ. γραμμικοί μηχανισμοί προσοχής που μειώνουν την πολυπλοκότητα από O(n^2) σε O(n), μπορούν να καταστήσουν εφικτή την πρόβλεψη ακόμα και σε δίκτυα δεκάδων χιλιάδων κόμβων με πολύ μικρό υπολογιστικό κόστος . Η διερεύνηση τέτοιων τεχνικών και η πιθανή ενσωμάτωσή τους στο STG-Tx επιτρέπει την εφαρμογή του μοντέλου αυτού και σε ακόμη μεγαλύτερης κλίμακας προβλήματα (π.χ. πανεθνικά δίκτυα αισθητήρων) ή/και σε περιβάλλοντα πραγματικού χρόνου με περιορισμένο hardware. Τρίτον, μια ενδιαφέρουσα κατεύθυνση είναι η ανάπτυξη μιας μετα-μαθησιακής προσέγγισης (transfer learning) όπου το μοντέλο εκπαιδευμένο στην Καλιφόρνια μπορεί να προσαρμοστεί (fine-tune) σε άλλη πόλη ή χώρα με μικρότερο νέο dataset. Το LargeST dataset προσφέρει τη δυνατότητα πειραματισμού σε αυτό, π.χ. θα μπορούσαμε να εκπαιδεύσουμε το STG-Tx στο πλήρες δίκτυο CA και μετά να το εξειδικεύσουμε με λίγα δεδομένα στη Νέα Υόρκη ή σε ευρωπαϊκές πόλεις, αξιοποιώντας τη γνώση που ήδη έχει (ότι π.χ. υπάρχουν ημερήσιοι κύκλοι, χωρικές συσχετίσεις κλπ.). Αυτό θα μείωνε το απαιτούμενο πλήθος δεδομένων για νέα περιοχές και θα έδειχνε κατά πόσο τα μοτίβα κυκλοφορίας είναι μεταφέρσιμα μεταξύ διαφορετικών αστικών περιβαλλόντων. Τέταρτον, θα μπορούσε να εξεταστεί η δυνατότητα online μάθησης: αντι να εκπαιδεύεται το μοντέλο εξ αρχής περιοδικά, να ενημερώνεται προοδευτικά με νέα δεδομένα (π.χ. κάθε μέρα ή εβδομάδα) ώστε να μαθαίνει τυχόν αλλαγές στις συνήθειες μετακίνησης ή στην υποδομή (λ.χ. αν άνοιξε ένας νέος δρόμος ή άλλαξε η κυκλοφορία λόγω τηλεργασίας). Αυτό βέβαια απαιτεί προσοχή για να μην αποσταθεροποιηθεί το εκπαιδευμένο μοντέλο, αλλά σύγχρονες τεχνικές συνεχούς μάθησης θα μπορούσαν να εφαρμοστούν.

Η δυνατότητα ακριβούς πρόβλεψης της κυκλοφορίας που προσφέρει το STG-Tx έχει άμεσες εφαρμογές σε διάφορα σενάρια ενός ευφυούς συστήματος μεταφορών. Ένα από τα πρωταρχικά πεδία αξιοποίησης είναι η δυναμική διαχείριση κυκλοφορίας από κέντρα ελέγχου. Γνωρίζοντας εκ των προτέρων ότι σε 30-60 λεπτά θα δημιουργηθεί συμφόρηση σε ένα συγκεκριμένο τμήμα του δικτύου, οι διαχειριστές μπορούν να λάβουν προληπτικά μέτρα: για παράδειγμα, να τροποποιήσουν τα προγράμματα φωτεινής σηματοδότησης στις γειτονικές αρτηρίες ώστε να εκτρέψουν μέρος της κίνησης, ώστε να ελέγξουν την ροή που εισέρχεται σε έναν ήδη φορτισμένο αυτοκινητόδρομο, ή να ανοίξουν προσωρινά λωρίδες έκτακτης ανάγκης. Επίσης, μπορούν να ενημερώσουν έγκαιρα τους οδηγούς μέσω μηνυμάτων σε ηλεκτρονικές πινακίδες (VMS) ή μέσω ραδιοφώνου/διαδικτύου ότι π.χ. «σε 20 λεπτά αναμένεται καθυστέρηση 15 λεπτών στην Εθνική Οδό – εξετάστε εναλλακτική διαδρομή». Έτσι, το πρόβλημα θα μπορούσε να μετριαστεί ή και να αποφευχθεί χάρη στην πρόγνωση, αντί η επέμβαση να γίνεται αφού ήδη δημιουργηθεί μποτιλιάρισμα.

Οι υπηρεσίες πλοήγησης αποτελούν έναν ακόμα τομέα εφαρμογής. Σήμερα αυτές οι υπηρεσίες βασίζονται κυρίως σε αντιδραστική πληροφόρηση – δηλαδή διαπιστώνουν την κυκλοφοριακή συμφόρηση αφού έχει συμβεί (μέσω των ταχυτήτων των οχημάτων των χρηστών) και κατόπιν αναπροσαρμόζουν διαδρομές. Εάν όμως ενσωματωθούν προβλέψεις όπως του STG-Tx, η καθοδήγηση μπορεί να γίνει προδραστική: το σύστημα πλοήγησης μπορεί να αποφύγει μια διαδρομή που αναμένεται να έχει καθυστέρηση σε λίγο, ακόμη κι αν την ώρα της εκκίνησης φαίνεται ομαλή. Αυτό θα διανείμει καλύτερα την κυκλοφορία και θα μειώσει το φαινόμενο όπου όλοι οι οδηγοί ακολουθούν μια συμβουλή που ήταν καλή εκείνη τη στιγμή αλλά γίνεται κακή 20 λεπτά μετά επειδή επιλέχθηκε μαζικά. Στον τομέα των εμπορευματικών μεταφορών και της εφοδιαστικής αλυσίδας (logistics), η πρόβλεψη κυκλοφορίας μπορεί να χρησιμοποιηθεί για βελτιστοποίηση δρομολογίων και δρομολόγησης στόλων. Για παράδειγμα, μια εταιρεία διανομών μπορεί το πρωί να σχεδιάσει δυναμικά τις παραδόσεις της λαμβάνοντας υπόψη τις προβλέψεις κυκλοφορίας σε όλη την πόλη για τις επόμενες ώρες, ώστε να αποφεύγει ζώνες και ώρες αιχμής όσο είναι δυνατόν. Αυτό οδηγεί σε εξοικονόμηση χρόνου και καυσίμων, με οικονομικό αλλά και περιβαλλοντικό όφελος.

Σε πιο στρατηγικό επίπεδο, τα ίδια τα μοτίβα που μαθαίνει και αναδεικνύει το μοντέλο μπορούν να καθοδηγήσουν τον σχεδιασμό υποδομών. Εάν παρατηρείται μέσω των προβλέψεων ότι συγκεκριμένοι δρόμοι έχουν τακτικά ζήτηση που υπερβαίνει την προσφορά (π.χ. η πρόβλεψη σχεδόν πάντα υποεκτιμά λίγο την πραγματικότητα επειδή υπάρχουν συχνά γεγονότα που προκαλούν περισσότερη κίνηση), αυτό υποδηλώνει ότι το οδικό δίκτυο εκεί είναι ευάλωτο ή κορεσμένο. Οι συγκοινωνιολόγοι μπορούν να χρησιμοποιήσουν τέτοια ευρήματα για να εντοπίσουν πού απαιτούνται επεμβάσεις: π.χ. αν ένα μοντέλο δείχνει συχνά ότι ακόμα και χωρίς ατύχημα προβλέπεται συμφορημένο ένα τμήμα, ίσως χρειάζεται διαπλάτυνση ή νέα εναλλακτική διαδρομή. Επίσης, μέσω προγνωστικών μοντέλων μπορούν να αξιολογηθούν σενάρια («τι θα συμβεί αν»): τι θα συμβεί αν κλείσει μια λωρίδα σε μια αρτηρία; Τι επίδραση θα έχει μια νέα γραμμή Μετρό στη ροή οχημάτων; Το εκπαιδευμένο STG-Tx, σε συνδυασμό με προσομοιώσεις, μπορεί να συμβάλει στην απάντηση τέτοιων ερωτημάτων, αποτελώντας εργαλείο υποστήριξης αποφάσεων για τις αρχές.

Τέλος, με τη ραγδαία άνοδο των διασυνδεδεμένων και αυτόνομων οχημάτων, ανοίγεται ένας νέος ορίζοντας όπου τα ίδια τα οχήματα θα μπορούσαν να αξιοποιούν τις προβλέψεις κυκλοφορίας. Ένα αυτόνομο όχημα που γνωρίζει ότι 5 χιλιόμετρα μπροστά του η ταχύτητα κυκλοφορίας θα μειωθεί δραστικά σε 10 λεπτά, μπορεί να προσαρμόσει το προφίλ οδήγησής του: να αρχίσει να μειώνει ταχύτητα νωρίτερα και πιο ομαλά, να επιλέξει εναλλακτική διαδρομή ή να μεταβάλει την πορεία του έτσι ώστε να αποφύγει έντονα φρεναρίσματα. Αυτό θα βελτιώσει τόσο την οδική ασφάλεια (λιγότερες πιθανότητες καραμπόλας από απότομη επιβράδυνση) όσο και την κατανάλωση καυσίμου/ενέργειας, καθώς η ομαλή ροή μειώνει τα απότομα ανεβοκατεβάσματα στροφών κινητήρα. Σε ένα πλήρως ευφυές σύστημα μεταφορών, όπου τα οχήματα, οι φωτεινοί σηματοδότες και οι υποδομές επικοινωνούν, οι προβλέψεις κυκλοφορίας θα είναι ένας από τους πυρήνες συντονισμού: θα επιτρέπουν σε όλα τα υποσυστήματα να συγχρονίζονται με βάση το τι πρόκειται να συμβεί, όχι μόνο τι συμβαίνει τώρα.

Συμπερασματικά, η παρούσα εργασία ανέπτυξε και αξιολόγησε το μοντέλο STG-Tx σε δεδομένα πρωτοφανούς μεγέθους (LargeST-CA) και απέδειξε ότι η συνεκτική χωροχρονική μάθηση μπορεί να αντιμετωπίσει τις προκλήσεις μεγάλης κλίμακας προσφέροντας κορυφαία ακρίβεια πρόβλεψης. Η σημασία της πρόβλεψης κυκλοφορίας σε πραγματικά σενάρια ITS καθιστά τέτοιου είδους μοντέλα πολύτιμα: από τη βελτιστοποίηση καθημερινών μετακινήσεων έως τον μακροπρόθεσμο σχεδιασμό υποδομών, η δυνατότητα να «βλέπουμε μπροστά» στην κυκλοφορία μεταφράζεται σε εξοικονόμηση χρόνου, κόστους και ζωών. Οι μελλοντικές επεκτάσεις που συζητήθηκαν μπορούν να κάνουν το STG-Tx ακόμη πιο ισχυρό και χρήσιμο, φέρνοντάς μας πιο κοντά σε πόλεις όπου η κυκλοφορία ρέει πιο ομαλά, πιο έξυπνα και πιο βιώσιμα χάρη στην επιστήμη των δεδομένων και της τεχνητής νοημοσύνης.

Λέξεις-Κλειδιά

STG-Tx, Spatio-Temporal Graph Transformer, LargeST-CA, πρόβλεψη κυκλοφορίας, χωροχρονικά γραφήματα, mixed precision training, node subsampling, μετασχηματιστές (Transformers), πρόβλεψη χρονοσειρών, ευφυή συστήματα μεταφορών

Chapter 1 - Introduction

1.1 Background

The rapid expansion of urbanization and motorization over the past decades has generated unprecedented pressures on transportation infrastructures worldwide. Cities and regions are confronted with chronic congestion, extended travel delays, unpredictable journey times, and elevated risks of traffic-related incidents. These challenges have motivated the evolution of Intelligent Transportation Systems (ITS), which aim to harness sensing technologies, communication platforms, and automated control strategies to improve both the operational efficiency and safety of road networks. At the heart of most ITS applications lies the ability to anticipate short-term traffic conditions with sufficient accuracy and timeliness to guide proactive interventions.

Short-term traffic forecasting generally refers to the prediction of core traffic variables—such as flow, speed, and occupancy—over horizons ranging from five minutes to one hour. Forecasts within this time scale are highly actionable: they can support adaptive signal control, ramp metering, and dynamic message signs, all of which can be adjusted to forestall congestion before it fully materializes. Forecasts also serve as essential inputs to real-time route guidance systems and incident management platforms, which can help reduce secondary collisions and recover network functionality more rapidly after disruptions. Studies have shown that deploying prediction-informed traffic management can reduce total travel delay and collision risks by more than fifteen percent in heavily loaded freeway corridors [1].

The pursuit of short-term traffic forecasting has produced a long trajectory of methodological innovation. The earliest methods emerged from the tradition of statistical time-series modeling. Among these, the Auto-Regressive Integrated Moving Average (ARIMA) model represented a seminal contribution. ARIMA assumes linearity and weak stationarity, which means it is well-suited to recurring, regular traffic patterns such as those observed on stable links with pronounced diurnal cycles [2]. Seasonal variants of ARIMA extended this capability to capture weekly periodicities. Another influential approach was the application of Kalman filters, which provide online recursive state estimation under Gaussian noise assumptions. Kalman filtering has been particularly attractive for freeway speed inference, where estimates must be updated in real time as new sensor data streams in. These models proved effective in constrained scenarios, yet they faced severe limitations when deployed on networks experiencing nonlinear dynamics, recurrent congestion waves, and spatial spill-back effects across adjacent road links. Moreover, their reliance on strong distributional assumptions restricted their ability to cope with missing or corrupted detector data.

During the early 2000s, the field began to integrate methods from machine learning, which enabled more flexible modeling of nonlinear patterns. Techniques such as Support Vector Regression (SVR) and Random Forests (RF) gained popularity. These methods no longer depended on explicit physical or linear assumptions and could instead fit nonlinear mappings directly from historical data. While they often outperformed classical statistical baselines, they suffered from two important weaknesses. First, they required substantial manual feature engineering, including the design of lagged features, polynomial expansions, or Fourier transforms to capture seasonality. Second, they largely treated each sensor independently, failing to exploit the rich spatial correlations inherent in traffic networks. In practice, this meant that while SVR or RF might yield improved accuracy at individual sensor sites, they could not capture the network-level interactions that drive congestion propagation.

The advent of deep learning in the mid-2010s represented a paradigm shift. Deep neural networks introduced the possibility of learning hierarchical feature representations directly from raw time series, reducing the need for manual feature engineering. Convolutional Neural Networks (CNNs) were first applied to traffic forecasting to learn localized temporal kernels, capturing short-term variations in flow or speed. Recurrent Neural Networks (RNNs), particularly the Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) variants, excelled at modeling longer dependencies, enabling forecasts that accounted for trends extending beyond a few minutes. These architectures quickly demonstrated superior performance over traditional machine learning and statistical methods, particularly under conditions of high variability.

Nevertheless, CNNs and RNNs alone could not capture the non-Euclidean spatial dependencies that characterize road networks. This led to the emergence of Graph Neural Networks (GNNs) in transportation research. GNNs operate directly on graph structures, leveraging adjacency information to propagate signals between nodes in a network. When combined with temporal sequence models, they gave rise to Spatio-Temporal Graph Neural Networks (STGNNs). These architectures represent one of the most influential breakthroughs in modern traffic forecasting, as they explicitly model both the graph topology of the road network and the temporal evolution of traffic states. Two landmark examples are the Diffusion Convolutional Recurrent Neural Network (DCRNN) [3], which integrates diffusion processes with recurrent units, and the Spatio-Temporal Graph Convolutional Network (STGCN) [4], which applies convolutional filters in both spatial and temporal domains. Both models demonstrated that incorporating non-Euclidean graph structures substantially improves accuracy compared to treating sensors independently or approximating road networks as Euclidean grids.

Building on these advances, researchers recently turned toward the Transformer architecture, first introduced for machine translation [5]. Transformers rely on global self-attention, a mechanism that allows every token in a sequence to attend to all others, capturing long-range dependencies without the recurrence constraints of RNNs. This property made transformers attractive for time-series forecasting, including traffic data, where dependencies often extend across hours or even days. Vanilla transformers, however, impose quadratic complexity in both memory and computation with respect to sequence length. This creates a major obstacle when working with traffic datasets, where input sequences can extend to hundreds of time steps across thousands of sensors.

Early adaptations of transformers for time series sought to mitigate this complexity. The Temporal Fusion Transformer (TFT) [6] incorporated gating and attention to improve interpretability and reduce redundancy, making it feasible for multivariate forecasting tasks. The Informer architecture [7] introduced sparse attention mechanisms, reducing complexity for long univariate or small multivariate sequences. In the traffic domain, further specialization followed. STAEformer [8] enhanced vanilla transformers with spatio-temporal adaptive embeddings, achieving state-of-the-art results on medium-sized networks. PDFormer extended this by integrating propagation-delay awareness, explicitly modeling the delayed effects of congestion spread. The most recent STGformer [9] collapsed spatial and temporal attention into a single layer, reporting substantial efficiency gains and demonstrating the potential of transformers for very large graphs. Collectively, these innovations underscore the promise of transformer models for traffic forecasting, but also highlight the unresolved challenge of applying them to truly statewide networks in real time.

1.2 Problem Context

Forecasting traffic at the scale of entire freeway systems involves challenges that go far beyond those encountered in smaller academic benchmarks. Real-world deployments such as California's freeway sensor network involve thousands of sensors streaming data continuously at high frequency. This scale creates unique technical hurdles that models must address to be operationally relevant.

The first major issue is dimensionality. With 8,600 sensors reporting every five minutes, each statewide snapshot represents thousands of correlated variables. Over the course of a single year, this amounts to approximately 8.1 billion individual values. Processing such vast datasets in real time is computationally expensive. Models trained on smaller datasets may simply not scale, both because of GPU memory limitations and because the learning algorithms themselves may not generalize well when input dimensionality increases by orders of magnitude.

The second issue is the non-Euclidean nature of road networks. Freeways and arterials form complex, directed graphs with highly irregular connectivity. Traditional CNNs, which assume grid-like Euclidean structures, are therefore unsuitable. For example, two freeway ramps may be physically adjacent but not directly connected in terms of traffic flow, while two distant segments may strongly interact due to network rerouting. Forecasting models must capture these graph-structured dependencies explicitly, which is why graph neural networks and attention mechanisms have become indispensable.

Third, traffic systems exhibit profound temporal heterogeneity. Daily commuting patterns create strong morning and evening peaks, while weekly rhythms distinguish weekdays from weekends. Superimposed on these are seasonal variations, such as increased holiday travel or reduced summer commuting. On top of these regular cycles are stochastic disruptions—incidents, roadworks, or adverse weather—that defy simple periodicity. Any model must be able to capture the interaction between deterministic rhythms and unpredictable disturbances.

Fourth, data quality remains an unavoidable problem. Inductive loop detectors, which form the backbone of most freeway monitoring systems, are prone to outages, sensor noise, or calibration drift. Missing-value rates of up to five percent per sensor are not uncommon [12]. Forecasting systems must therefore incorporate strategies for robust handling of missing and noisy data, without compromising accuracy.

Traditional research datasets have only partially captured these challenges. Popular benchmarks such as PeMSD4 (307 sensors, two months), METR-LA (207 sensors, four months), and PeMSD8 (170 sensors, two months) have proven invaluable for developing and comparing models. However, they represent only small subgraphs of the California freeway system, and they span relatively short time horizons. Models trained and tuned on such small datasets may achieve high reported accuracy but risk poor generalization when scaled to thousands of nodes and multi-year horizons. In practice, models must be validated under realistic conditions that reflect the full complexity of statewide operations. Without this, the research-to-practice gap remains significant.

1.3 Motivation

The LargeST-CA corpus was introduced to close this scale gap by aggregating five years (2017–2021) of loop-detector data at 5-minute granularity, spanning the entire California highway system [10]. This dataset comprises approximately 4.5×10^9 individual observations and includes a graph with 8,600 nodes and 201,000 edges, making it the most comprehensive publicly available benchmark for large-scale traffic forecasting. By encompassing both dense urban freeways and sparsely instrumented rural highways, LargeST-CA presents a uniquely realistic challenge for forecasting models.

However, applying existing state-of-the-art STGNNs and transformer variants naively to this dataset exposes severe scalability bottlenecks. For example, GNN + RNN hybrids such as DCRNN [3] and STGCN [4] require per-time-step graph convolutions combined with gated recurrences. On LargeST-CA, a 24-hour input window (288 steps) implies on the order of 200 GFLOPs per forward pass and demands more than 40 GB of GPU memory, already exceeding the capacity of many commodity GPUs.

Similarly, early transformer adaptations are not immediately applicable. STAEformer [8] stacks separate temporal and spatial attention layers, each of which scales quadratically with sequence length or node count. For 8,600 nodes across 288 time steps, the memory requirements explode into the terabyte scale, which is infeasible even on modern datacenter GPUs. PDFormer, while alleviating some temporal shift issues by incorporating propagation delay awareness, still relies on deep attention stacks that remain unsuitable for statewide graphs.

The recently proposed STGformer [9] introduced a clever compression by collapsing spatial-temporal interactions into a single linearized attention layer, reporting up to a 100× inference speed-up compared to STAEformer. However, its implementation relies on custom CUDA kernels and large embedding stacks, which limit its modularity and ease of integration. This raises barriers for reproducibility and extension by other researchers.

These limitations motivate the development of a new model: the Spatio-Temporal Graph Transformer (STG-Tx). The goal of STG-Tx is to achieve three main design principles. First, to decouple temporal and spatial computation into separate modules, promoting transparency and interpretability in how the model handles time and space. Second, to apply patch-wise compression to bound the length of attention computations, thereby reducing memory requirements while retaining coverage of key patterns. Third, to leverage FlashAttention kernels to implement memory-linear self-attention, enabling the model to handle long input sequences and large node sets within realistic GPU budgets.

1.4 Objectives and Scope

Building on the background and motivation, this thesis pursues five interrelated objectives:

Architecture design: The first objective is to design and implement the dual-path STG-Tx architecture. This involves integrating a temporal patch attention mechanism with a spatial graph attention mechanism, fused through a lightweight gating component. The architecture aims to strike a balance between expressive power and computational efficiency, targeting statewide networks rather than small laboratory graphs.

End-to-end pipeline: The second objective is to develop a GPU-accelerated workflow for data ingestion, storage, feature engineering, training, and evaluation. This includes the use of Arrow/Parquet storage formats for scalable handling of raw detector data, and NVTabular for efficient preprocessing. The workflow ensures that LargeST-CA can be processed reproducibly at scale, providing a template for future large traffic datasets.

Accuracy assessment: The third objective is to rigorously evaluate the predictive quality of STG-Tx using standard error metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). Performance is measured across 12 forecast horizons (5 to 60 minutes), enabling fine-grained analysis of short- and medium-term predictive accuracy.

Efficiency benchmarking: The fourth objective is to measure the efficiency of STG-Tx in practice, including wall-clock training time, inference latency, peak GPU memory consumption, and monetary cost when deployed on cloud GPUs such as the AWS H100. Comparisons are drawn against established baselines—DCRNN, STGCN, STAEformer, PDFormer, and STGformer—allowing us to quantify trade-offs between accuracy and efficiency.

Analytical insight: The fifth objective is to provide a detailed analysis of strengths, error modes, and scalability limits of STG-Tx. This includes examining which traffic regimes are forecasted well (e.g., recurrent peak periods) and which remain challenging (e.g., rare incidents). Future directions such as adaptive edge weighting, multi-modal data fusion, and continual learning are outlined, situating this work within a broader research agenda.

Scope boundaries. The study focuses exclusively on forecasting traffic flow variables derived from inductive loop detectors. Other data modalities such as travel times from GPS, public transport feeds, pedestrian counts, or microscopic traffic simulation outputs are beyond the scope of this thesis. By constraining the problem definition, we ensure that evaluation remains tractable while directly addressing the most widely deployed sensing infrastructure in freeway networks.

1.5 Thesis Structure

Chapter 2 surveys classical, machine-learning, and deep spatio-temporal forecasting methods, highlighting scalability and efficiency gaps.

Chapter 3 introduces the LargeST-CA dataset, the GPU preprocessing pipeline, and the proposed STG-Tx architecture.

Chapter 4 describes implementation choices, including training on a Paperspace A100 GPU, mixed-precision arithmetic, node subsampling, and reproducibility tooling.

Chapter 5 presents experimental results, reporting MAE ≈ 30.2, RMSE ≈ 39.7, and MAPE ≈ 8.3% on the held-out test set, and analyzing the impact of efficiency techniques.

Chapter 6 discusses findings, strengths, limitations, and deployment implications for real-world ITS.

Chapter 7 concludes the thesis and outlines future work, including forecast-driven traffic signal control, multimodal fusion, and real-time adaptation.

Summary

Modern ITS demand forecasts that scale to state-level sensor networks while maintaining real-time latency. Existing deep models either sacrifice efficiency or limit scope to small graphs. Leveraging the LargeST-CA benchmark, this thesis introduces STG-Tx, an efficient Spatio-Temporal Graph Transformer that combines patch-wise temporal attention, adaptive graph attention, and FlashAttention acceleration. The ensuing chapters build a coherent narrative from literature gaps to algorithmic design, large-scale evaluation, and critical appraisal, thereby contributing a scalable blueprint for next-generation traffic forecasting.

References

[1] H. Dia, “An agent-based approach to modelling driver route choice behaviour under the influence of real-time information systems,” Transportation Research C, vol. 10, no. 5–6, pp. 331–349, 2002.
[2] B. Williams and L. Hoel, “Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results,” J. Transp. Eng., vol. 129, no. 6, pp. 664–672, 2003.
[3] Y. Li, R. Yu, C. Shahabi, and Y. Liu, “Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting,” Proc. ICLR, 2018, arXiv:1707.01926.
[4] B. Yu, H. Yin, and Z. Zhu, “Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting,” Proc. IJCAI, 2018, arXiv:1709.04875.
[5] A. Vaswani et al., “Attention Is All You Need,” Adv. NeurIPS, vol. 30, 2017.
[6] B. Lim et al., “Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting,” Adv. NeurIPS, vol. 34, pp. 17436–17447, 2021.
[7] H. Zhou et al., “Informer: Beyond Efficient Transformers for Long Sequence Time-Series Forecasting,” Proc. AAAI, vol. 35, no. 12, pp. 11106–11115, 2021.
[8] Y. He et al., “STAEformer: Spatio-Temporal Adaptive Embedding Makes Vanilla Transformer SOTA for Traffic Forecasting,” arXiv:2308.10425.
[9] H. Wang et al., “STGformer: Efficient Spatiotemporal Graph Transformer for Traffic Forecasting,” arXiv:2410.00385, 2024.
[10] X. Liu et al., “LargeST: A Benchmark Dataset for Large-Scale Traffic Forecasting,” NeurIPS Datasets & Benchmarks, 2023, arXiv:2306.08259.

Chapter 2 - Traffic-Forecasting Methods

2.1.1 An overview

Early traffic prediction research was grounded in time-series statistics, exploiting the repetitive nature of flows on individual road segments. The simplest baselines—historical mean, median, or seasonal naïve methods—assume that conditions observed at the same clock time on previous days will reoccur; despite negligible computational cost, mean absolute errors (MAE) often exceed 20 % under incident or holiday regimes. Exponential smoothing adds a memory factor to adapt gradually to trend shifts but still presumes linear evolution [1].

The seminal work of Ahmed & Cook applied the Box–Jenkins Auto-Regressive Integrated Moving Average (ARIMA) framework to freeway loop-detector series, demonstrating that a well-fitted ARIMA model could reduce root-mean-square error (RMSE) by ≈12 % relative to a seasonal mean on 15-min ahead forecasts [2]. Seasonal extensions (SARIMA) incorporate daily or weekly differencing and have been reported to improve short-horizon speed prediction on urban arterials.

Kalman filtering formulates flow dynamics in a state-space model and updates estimates on line with Gaussian noise assumptions. Okutani & Stephanedes pioneered its use for dynamic origin–destination estimation, achieving sub-5 % relative error on a 10-link corridor [3]. Variants such as the Extended and Unscented Kalman Filters handle mild nonlinearities but still require manual tuning of process-noise covariance matrices.

Strengths: Classical models are interpretable, grounded in statistical theory, and trivial to implement in SQL or embedded controllers. They require few parameters, facilitating deployment on low-power hardware.

Limitations: Their linearity and stationarity assumptions break down under regime shifts (e.g., incidents, weather). Most implementations treat each detector independently, ignoring spatial spill-back. Parameter calibration scales linearly with node count N, making statewide maintenance impractical when N>5 000. Consequently, by the late 1990s their performance plateaued, motivating the adoption of non-linear machine-learning models.

2.1.2 Shallow Machine-Learning Models

Non-linear Regressors: The mid-1990s saw the introduction of Support Vector Regression (SVR), which fits an ε-insensitive loss in a high-dimensional kernel space. Drucker et al. reported that SVR lowered MAE by 15 % over linear regression on Boston freeway speeds, at the expense of solving a quadratic program whose complexity grows with training size [4].

Decision-Tree Ensembles: Ensemble decision trees, notably Random Forests (RF) and Gradient Boosting Machines (GBM), gained traction owing to their ability to capture feature interactions without heavy tuning. Breiman's RF algorithm averaged 500 trees and cut RMSE by 18 % compared with SVR on a two-month PeMS sample [5].

Feature Engineering & Scalability Issues: Shallow learners rely on manual feature engineering: lagged flows, day-of-week dummies, weather, event indicators, and sometimes upstream sensor readings. Such pipelines demand domain expertise and can miss nonlinear spatial dependencies. Computationally, per-sensor model training is feasible for networks with N≤500 but becomes onerous for statewide deployments.

2.1.3 Emergence of Deep Learning

Early Deep Feed-Forward Networks. With greater data availability, researchers explored Multilayer Perceptrons (MLP) and stacked auto-encoders. Huang et al. achieved a 7 % MAE reduction versus SVR on Beijing ring-road flows using a 5-layer auto-encoder that autonomously learned temporal abstractions [6]. Despite improvements, these fully-connected architectures disregarded spatial context and required fixed-length inputs, hindering long-horizon forecasting.

To capture local spatial correlations, Convolutional Neural Networks were applied to grid-mapped traffic measures. Zhang, Zheng & Qi's DeepST model represented citywide taxi demand as an m×n image and stacked residual CNNs with a parametric fusion of daily, weekly, and monthly trends, lowering RMSE by 17 % on New York demand heat-maps [7]. For sensor networks, CNN–LSTM hybrids treated lagged measurements as channels; a 2016 California study reported a 12 % MAE gain over RF using a three-layer CNN followed by a bidirectional LSTM [8]. While hybrids improved temporal learning, the Euclidean grid assumption forced artificial interpolation between irregular sensor locations.

Graph-Structured Deep Networks. The breakthrough for large-scale traffic forecasting arrived with Spatio-Temporal Graph Neural Networks (STGNNs), which natively process non-Euclidean graphs.

- Diffusion Convolutional Recurrent Neural Network (DCRNN) introduces a bidirectional diffusion operator T^k that propagates information through k-hop neighbours and embeds it into a GRU encoder–decoder [9]. On the 207-sensor METR-LA dataset, DCRNN reduced MAE by 12 % and RMSE by 15 % relative to LSTM baselines.

- Spatio-Temporal Graph Convolutional Network (STGCN) replaces recurrent units with Gated Temporal Convolutions and Chebyshev graph kernels to parallelise training, trimming epoch time by 80 % while achieving comparable accuracy on PeMSD7 (883 sensors) [10].

These architectures inaugurated the current research line in which spatial graph convolutions (GCN, GraphSAGE, Diffusion) are combined with temporal modules (RNN, TCN, self-attention) to model complex propagation phenomena such as queue spill-back and shock-waves. Their success also exposed computational bottlenecks—quadratic growth of attention matrices and increasing depth—when extending from city-scale graphs (<1 000 nodes) to statewide networks (>8 000 nodes), a limitation addressed in Section 2.2.

2.1.4 Summary

Traffic-forecasting methodology has evolved from statistically grounded yet spatially myopic ARIMA/Kalman models, through shallow non-linear regressors requiring heavy feature engineering, to deep spatio-temporal networks that learn representations directly from data. Classical and shallow techniques remain attractive for single-link dashboards, but deep networks now dominate benchmark leaderboards due to their ability to capture multivariate nonlinear dynamics and graph topology. Nevertheless, existing deep models struggle with scalability and memory efficiency on truly large graphs, motivating the exploration of more compact graph-attention architectures discussed in Section 2.2.

References

[1] E.S. Gardner Jr, “Exponential Smoothing: The State of the Art,” Journal of Forecasting, vol. 4, no. 1, pp. 1–28, 1985.
[2] M. S. Ahmed and A. R. Cook, “Analysis of Freeway Traffic Time-Series Data by Using Box-Jenkins Techniques,” Transportation Research, vol. 13, no. 1, pp. 1–9, 1979.
[3] S. Okutani and Y. Stephanedes, “Dynamic Prediction of Traffic Volume Through Kalman Filtering Theory,” Transportation Research B, vol. 18, no. 1, pp. 1–11, 1984.
[4] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik, “Support Vector Regression Machines,” Adv. NIPS, vol. 9, 1996.
[5] L. Breiman, “Random Forests,” Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
[6] W. Huang, G. Song, H. Hong, and K. Xie, “Deep Architecture for Traffic Flow Prediction: A Deep Belief Network,” IEEE T-ITS, vol. 15, no. 5, pp. 2191–2201, 2014.
[7] J. Zhang, Y. Zheng, and D. Qi, “Deep Spatio-Temporal Residual Networks for Citywide Crowd Prediction,” arXiv:1610.00081.
[8] X. Ma et al., “Long Short-Term Memory Neural Network for Traffic Speed Prediction Using Remote Microwave Sensor Data,” Transportation Research C, vol. 54, pp. 187–197, 2015.
[9] Y. Li, R. Yu, C. Shahabi, and Y. Liu, “Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting,” Proc. ICLR, 2018, arXiv:1707.01926.
[10] B. Yu, H. Yin, and Z. Zhu, “Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting,” Proc. IJCAI, 2018, arXiv:1709.04875.

2.2 Graph Neural Networks & Hybrid Deep Models

2.2.1 Graph-Based Spatial Modelling

Urban road-sensor networks form a non-Euclidean domain: each detector (node) is anchored on a directed road segment and interacts with upstream or downstream links through topological, directional, and distance-weighted edges. Classical convolutions defined on regular grids cannot capture such irregular connectivity, motivating the use of graph signal processing. Graph modelling enables (i) localisation of congestion propagation along realistic paths, (ii) information sharing across non-adjacent yet highly correlated corridors (e.g., freeway detours), and (iii) principled handling of missing or partially observed sensors by message passing. However, road graphs are dynamic: lane closures, incidents, and time-of-day signal plans alter effective edge weights, rendering a static adjacency matrix sub-optimal. These observations underpin the design of Spatio-Temporal Graph Neural Networks (STGNNs) reviewed below.

2.2.2 Spatio-Temporal Graph Neural Networks (STGNNs)

STGNNs interleave graph-based spatial operators with sequence models that capture temporal evolution. Let G=(V,E) denote a sensor graph with |V|=N nodes, weighted adjacency A, and traffic features X∈ℝ<sup>N×T</sup> over a horizon T.

(a) Diffusion Convolutional Recurrent Neural Network (DCRNN) [1]

- Architecture: DCRNN models traffic as a bidirectional diffusion process. For each layer,

$$\mathrm{DiffConv}(X) = \sum_{k=0}^{K} \left[\theta_k^{(f)}(D^{-1}A)^k + \theta_k^{(b)}(D^{-1}A^\top)^k\right] X\$$$

  which is inserted into an encoder–decoder GRU with scheduled sampling.

- Innovation: Multi-hop diffusion captures upstream and downstream influence without hand-crafted lag features.

- Reported gain:. On METR-LA (207 sensors) and PEMS-BAY (325 sensors) DCRNN reduced RMSE and MAE by 12–15 % over ARIMA and Feed-Forward baselines [1].

- Limitations: Quadratic parameter growth in K-hop matrices plus recurrent unrolling hinders scaling beyond ≈ 1 000 sensors.

(b) Spatio-Temporal Graph Convolutional Network (STGCN) [2]

- Architecture: Stacks Chebyshev graph convolutions with gated temporal CNN (TCN) blocks, eliminating RNN recurrence.

- Innovation: Causal dilations in the TCN enable parallel GPU training; graph convolutions share weights across time.
- Performance: STGCN improved RMSE by ≈10 % over LSTM on METR-LA while training 10 × faster than DCRNN due to 1-D convolutions [2].
- Limitations: The model still multiplies dense (N×N) filters each layer; depth must grow to enlarge receptive field, inflating memory.

(c) Graph WaveNet (GWN) [3]

- Architecture: Introduces adaptive adjacency learned via node embeddings and dilated causal convolutions for time.
- Innovation: The learned adjacency uncovers hidden correlations (e.g., parallel arterials) without a pre-defined graph.
- Performance: On METR-LA, GWN achieved a 4 % lower MAE than STGCN and halved inference latency by avoiding RNNs [3].
- Limitations: Adaptive matrices are still N×N; memory rises quadratically for 8 k+ sensors.

(d) Attention-based STGCN (ASTGCN) [4]

- Architecture: Adds temporal and spatial attention gates atop STGCN blocks; weights are learned per node and per time step.
- Innovation: Attention improves interpretability—e.g., sensors near freeway ramps gain higher weights during peak hours.
- Performance: Achieved additional 2–4 % MAE reduction vs. GWN on Beijing and PeMSD7 datasets [4].
- Limitations: More parameters and attention maps further escalate $O(N^2)$ memory.

(e) Supporting Evidence of the Trend

- T-GCN combines a single GCN with a GRU, cutting MAE by ~5 % over pure GRU on Shenzhen taxi flows [5].
- AGCRN removes fixed graphs entirely via Node-Adaptive Parameter Learning, outperforming GWN by 6–8 % MAE on PeMSD3 with fewer parameters [6].
- MTGNN learns graph structure jointly with dilated inception TCNs, beating LSTNet by up to 18 % on four public datasets and winning KDD 2020 best-paper runner-up [7].

Together, these studies cement the GCN + temporal-block blueprint as the de-facto recipe for traffic forecasting circa 2018–2022.

2.2.3 Impact on Prediction Accuracy

Meta-analyses in recent surveys report 5–20 % lower MAE/RMSE for STGNNs versus classical or shallow baselines across 15+ benchmarks [10]. The gains stem from two synergistic factors:

1. Spatial correlations. Graph convolutions let each node aggregate upstream conditions (queue spill-back), downstream capacity (bottlenecks), and cross-network detours, capturing non-local influence that ARIMA or SVR cannot model.

2. Temporal dynamics. RNN/TCN blocks capture daily and weekly cycles plus gradual build-ups, while attention layers highlight irregular events (accidents, concerts) by assigning higher weights to rarer contexts.

2.2.4 Scalability & Complexity Limitations

Cost driver	Mathematical growth	Bottleneck on 8 600-sensor LargeST-CA
GCN layers	Each layer multiplies $A\in\mathbb{R}^{N\times N}$ with features ⇒ $O(N^2)$ flops + memory.	Dense A (≈74 M entries) demands >2 GB per layer in FP32; stacking 10 layers exceeds 20 GB even before activations.
Recurrent stacks (GRU/LSTM)	Sequential unrolling prevents temporal parallelism; hidden state $O(N·H)$.	Training time scales linearly with horizon; a 24 h input (288 steps) forces hour-long GPU epochs.
Multi-hop diffusion	K-hop diffusion duplicates K sparse matrices; memory O(K·	E
Dense attention in STGNN-Transformer hybrids	Spatial-attention: $O(N^2)$; Temporal-attention: $O(T^2)$.	Vanilla transformer on a 7 d window ($T = 2016$) allocates 1.2 TB attention scores—impossible on current GPUs.

Empirically, the public STGNN codebases crash on GPU memory when N>2 000 without aggressive batching or node sampling. Researchers often resort to graph partitioning, which disrupts long-range spatial coherence, or shorten sequences to ≤12 h, losing weekly periodicity. Such compromises motivated the design of Spatio-Temporal Graph Transformers (STGTs) that collapse spatial and temporal reasoning into one attention kernel with linear complexity, culminating in STGformer—the first model to process all 8 600 LargeST sensors on a single A100 card while achieving a 100 × inference speed-up and 99.8 % memory cut over STAEformer [8]. Complementarily, the FlashAttention kernel reduces temporal attention I/O, yielding 2–3 × wall-clock speed-ups on sequences of 1 k tokens or more [9].

2.2.5 Summary

In summary, STGNNs established the importance of joint spatial–temporal learning and propelled traffic-forecast accuracy to new heights, yet their quadratic complexity and stack-depth recursion erect barriers at state-wide scale. The next generation of models—graph transformers with IO-aware attention kernels—promise to preserve those accuracy gains while delivering sub-quadratic memory and runtime. Section 2.3 introduces these scalable architectures, setting the technical foundation for the STG-Tx model advanced in this thesis.

References

[1] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting," Proc. ICLR, 2018, arXiv:1707.01926.
[2] B. Yu, H. Yin, and Z. Zhu, "Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting," Proc. IJCAI, 2018, arXiv:1709.04875.
[3] Z. Wu et al., "Graph WaveNet for Deep Spatial-Temporal Graph Modeling," Proc. IJCAI, 2019, arXiv:1906.00121.
[4] S. Guo et al., "Attention-Based Spatio-Temporal Graph Convolutional Networks for Traffic Flow Forecasting," Proc. AAAI, 2019.
[5] L. Zhao et al., "T-GCN: A Temporal Graph Convolutional Network for Traffic Prediction," IEEE T-ITS, 2020, arXiv:1811.05320.
[6] L. Bai et al., "Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting," Proc. AAAI, 2021, arXiv:2007.02842.
[7] Z. Wu et al., "Connecting the Dots: Multivariate Time Series Forecasting with Graph Neural Networks," Proc. KDD, 2020, arXiv:2005.11650.
[8] H. Wang et al., "STGformer: Efficient Spatiotemporal Graph Transformer for Traffic Forecasting," arXiv preprint, 2024, arXiv:2410.00385.
[9] T. Dao et al., "FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness," Proc. NeurIPS, 2022, arXiv:2205.14135.
[10] W. Jiang and J. Luo, "Graph Neural Network for Traffic Forecasting: A Survey," Expert Systems with Applications, vol. 213, 2023.

2.3 Transformer-Based Models for Traffic Forecasting

Transformers in Time-Series Forecasting

Transformer architectures, rely on multi-head self-attention to capture dependencies without recurrence. In self-attention, each input is linearly projected to query, key, and value vectors, and an attention weight matrix is computed as:

$$A(Q,K,V) = \mathrm{softmax}\Big(\frac{QK^T}{\sqrt{d_k}}\Big)V$$

where d_k is the key dimension [1]. This mechanism allows the model to attend to relevant positions regardless of distance, addressing the limitations of RNNs and CNNs in modeling long-range temporal dependencies. Unlike RNNs (which propagate information sequentially and can struggle with long-term context) or CNNs (which have locality-limited receptive fields), a Transformer can relate any two time steps in O(1) pairwise operations (constant w.r.t. sequence distance). This yields superior parallelism and the ability to capture global patterns, critical for long time series where distant events influence future outcomes.

To apply Transformers to time-series forecasting, several adaptations have been proposed to handle long input horizons and inherent temporal patterns. Temporal Fusion Transformer (TFT) combines recurrent layers with self-attention for multi-horizon forecasting [2]. TFT uses an LSTM encoder for local temporality and a multi-head attention decoder for long-term dependencies, along with gating mechanisms to skip irrelevant features. Specialized components (Variable Selection Networks and gating layers) provide interpretability by quantifying each feature's importance, making TFT's decisions more transparent. Empirically, TFT demonstrated state-of-the-art accuracy on several benchmarks (electricity, traffic, retail) and yielded significant error reductions over previous RNN/CNN models, all while offering insights via attention weights and feature importances. One limitation of TFT is its hybrid complexity (it still relies on RNNs internally), which can be less efficient on very long sequences despite the attention component.

To improve efficiency for long sequences, researchers introduced sparse attention and series decomposition in Transformers. Informer introduced a ProbSparse self-attention mechanism that selects a subset of dominant queries for full attention, reducing complexity from $O(L^2)$ to $O(L \log L)$ for sequence length L [3]. It also employs self-attention distilling, iteratively reducing sequence length at each encoder layer, and a generative decoder that predicts all future points in one forward pass. These innovations allow Informer to handle very long inputs (e.g. 10k time steps) with much lower memory and time overhead. On large-scale benchmarks, Informer significantly outperformed previous Transformer variants and RNN-based models in long sequence forecasting [3], validating that Transformers can effectively model long-range temporal patterns when suitably optimized.

However, Informer's sparsity may neglect some finer context, and its performance can degrade if important but "sparsified" interactions are omitted (a potential trade-off for efficiency).

Another notable variant is Autoformer, which moves beyond ad-hoc sparse attention by integrating a series decomposition directly into the architecture [4]. Autoformer separates each time series into trend and seasonal components through an inner decomposition block, and it introduces an Auto-Correlation mechanism to replace dot-product attention for capturing periodic dependencies. By focusing on sub-series periodicity, the model can learn repeating seasonal patterns more effectively than vanilla attention. This design was shown to improve long-term forecast accuracy substantially – Autoformer achieved about 38 % lower error than prior Transformers on long-horizon benchmarks across diverse domains (energy, traffic, weather, etc.) [4]. It also improves efficiency, since the auto-correlation mechanism scales more gracefully and avoids quadratic attention on full sequences. The main advantage of Autoformer is its ability to model complex seasonal trends explicitly, though it assumes the time-series are amenable to clear seasonal-trend separation. In summary, these adaptations (TFT, Informer, Autoformer) demonstrate how vanilla Transformers can be modified for time-series forecasting, each contributing unique ideas (hybrid RNN-attention architecture, sparse attention and distillation, series decomposition) to improve forecasting accuracy and efficiency. Each model reports state-of-the-art results in its context, but they are typically evaluated on univariate or small multivariate datasets and do not incorporate spatial dependencies, which are crucial in traffic forecasting.

Spatio-Temporal Transformers for Traffic Forecasting

Applying Transformers to traffic data is challenging because one must capture not only temporal patterns but also spatial correlations defined by the road network graph. Traffic sensors are nodes on a graph with roads as edges, so models must handle graph-structured data where distant nodes (in index) may be physically connected and nearby. Naively applying a Transformer to all sensor time series ignores the known adjacency structure – the self-attention might learn some spatial relations implicitly, but it does not inherently know which sensors are neighbors. This can lead to inefficient learning and the need for enormous data to infer relationships that are already known from the road network. Hence, spatio-temporal Transformer models have emerged, integrating graph neural network (GNN) ideas or spatial attention mechanisms to better incorporate the topology of traffic systems. Key difficulties include: how to encode the graph connectivity or distances between sensors into the Transformer, how to handle potentially thousands of nodes (which makes full attention $O(N^2)$ in the number of sensors), and how to model dynamic spatial dependencies (traffic conditions propagate through the network over time with delays).

One line of research couples GNN layers with Transformer layers. For example, GMAN (Graph Multi-Attention Network, AAAI 2020) employed an encoder-decoder architecture with multiple spatio-temporal attention blocks to predict future traffic flow. Each block in GMAN used a spatial attention sub-layer (attending over road graph neighbors) followed by a temporal attention sub-layer, along with a gated fusion mechanism. This design showed that attention mechanisms can improve performance when combined with graph convolutions, and GMAN achieved competitive results on city traffic speed datasets. However, GMAN's multi-step attention was computationally heavy and still limited to relatively small networks (e.g. Los Angeles (207 sensors) or Beijing (City) scenarios). Similarly, an early ST-Transformer model proposed to separate spatial and temporal attention: it performed temporal self-attention on each sensor's time series and spatial self-attention on each time slice, in alternating fashion. This factorized attention approach reduces computational load compared to a fully joint attention over all nodes and times, and it improved over pure GNN baselines. Yet, a limitation noted was insufficient adaptability to dynamic traffic changes – a static separation of time and space attention cannot easily capture changing traffic routes (e.g. rerouting due to incidents) since spatial attention was confined to fixed neighbor relationships at each step.

To address dynamic connectivity and long-range interactions, recent transformer-based models explicitly incorporate road network knowledge into the attention mechanism. STAEformer is a Spatio-Temporal Adaptive Embedding Transformer that augments the vanilla Transformer with learned spatial embeddings for each sensor [5]. Essentially, STAEformer learns a continuous vector representation of each node's position in the network (adaptive to the data) and adds these embeddings into the attention computation, so that sensors with similar embedding (i.e. similar traffic patterns or connectivity) attend to each other more strongly. This approach allowed a "vanilla" Transformer model (with separate temporal and spatial attention layers) to achieve state-of-the-art forecasting accuracy on standard traffic benchmarks [5]. However, the model relies on stacking multiple attention layers (spatial and temporal) to capture higher-order interactions. This deep attention stack makes its computational complexity grow quadratically with the sequence length and linearly with number of layers, leading to very high memory usage for long sequences.

Another state-of-the-art model is PDFormer, short for Propagation Delay-aware Transformer. PDFormer explicitly accounts for the travel time delay of traffic information between sensors. It introduces a delay-aware attention mechanism: when attending to another node's time series, the model incorporates a learnable time-delay factor that aligns phases of traffic waves. For example, if an upstream sensor sees congestion, PDFormer's attention to a downstream sensor will be shifted by the estimated propagation time along the road, effectively attending to the appropriate lagged time step. This innovation enables modeling both short-range and long-range spatial dependencies dynamically, reflecting how traffic congestion propagates through the network. PDFormer achieved strong performance on medium-sized traffic graphs (e.g. PeMS-Bay, METR-LA), outperforming earlier GCN-based models. Nonetheless, its authors noted that multi-scale spatial fusion was still a challenge – PDFormer focuses on dynamic delays but doesn't explicitly integrate multi-resolution spatial features. In addition, like other Transformers, PDFormer's complexity can become burdensome as the network size grows.

Beyond these, several supporting works have explored attention mechanisms for traffic. GMFormer, AGTNet, and others extend Transformers with graph modules or adaptive gating, aiming to combine the strengths of GNNs and Transformers. For instance, some approaches employ an adaptive graph learning module to update the adjacency matrix in tandem with attention (so the model can infer new links or changing importance of roads over time), while others incorporate external factors (e.g. events, weather) through additional attention heads. These models are generally evaluated on metropolitan-scale data (hundreds of sensors). They report incremental improvements, but often at the cost of added complexity. In summary, transformer-based traffic models to date have demonstrated the effectiveness of attention for capturing complex spatio-temporal interactions, consistently achieving top accuracy on common benchmarks. However, most were developed and validated on relatively small graphs (typically <1 000 sensors). Their computational costs scale poorly, and they often assume fixed or static spatial relations. This leaves open the question of how to scale Transformers to large, real-world traffic networks with many thousands of nodes, which is the focus of recent research.

Recent Advances in Scaling to Large Networks

To overcome the scalability issues, researchers have proposed architectures that drastically reduce the computational burden of Transformers for very large sensor graphs. A breakthrough in this direction is STGformer, an efficient Spatio-Temporal Graph Transformer designed for the full-scale California traffic network (≈8.6k sensors) [7]. STGformer introduces a single-layer STG attention block that jointly captures spatial and temporal dependencies, instead of stacking many separate attention layers. In this block, the temporal and spatial dimensions are treated together by constructing queries, keys, and values that contain both time and node information. In effect, each sensor-time pair can attend to any other sensor-time pair in one pass, achieving high-order interactions in one layer. Crucially, STGformer adopts linear attention techniques – rather than using the standard softmax attention (which is quadratic in the number of queries and keys), it employs a kernel-based approximation that reduces attention complexity to $O(n)$ (where n is number of nodes times time steps). By replacing the softmax with a feature map inner product, STGformer avoids computing the full $n \times n$ attention matrix. This dramatically lowers memory usage and computation for large n, at a slight cost of not being an exact attention (though practically with negligible loss in accuracy for forecasting).

STGformer balances graph convolution and Transformer attention – it leverages known graph structure to limit unnecessary pairs and uses attention to capture global patterns beyond local neighbors. Experiments on the newly introduced LargeST benchmark (which contains 5 years of data for 8 600 California sensors) showed that STGformer outperforms prior Transformer models (including PDFormer and STAEformer) in forecasting accuracy. More remarkably, STGformer achieves these results with vastly improved efficiency: it is about 100× faster and uses 99.8 % less GPU memory than STAEformer during inference on the full California graph. In real terms, an inference batch that would exhaust a high-memory GPU with earlier models can run on STGformer with only ~0.2 % of the memory footprint. This massive gain comes from the one-layer design and linearized attention. Even training time and convergence improved, likely due to fewer layers (thus avoiding unstable deep attention) and better utilization of global information in one shot. The success of STGformer demonstrates that simplifying the Transformer architecture – by collapsing depth and using more efficient attention – can preserve predictive power while making large-scale training feasible. The authors report that STGformer, with only 0.7 million parameters, matched or exceeded the accuracy of STAEformer with 4.7 million parameters, underscoring a key insight: in traffic forecasting, a leaner model that fully exploits both spatial and temporal data may generalize better and train faster than an over-parameterized one.

Building on these ideas, our work implements a further simplified Transformer variant called STG-Tx (a streamlined version of STGformer) in the “largest graph” case study. The goal is to scale to the complete California network with minimal resources while retaining accuracy. The STG-Tx architecture uses a dual-path attention block per layer, consisting of a temporal-first attention followed by a spatial attention. Instead of joint spatio-temporal attention in one operation, STG-Tx factorizes it in two steps (similar to ST-Transformer’s strategy but improved with modern attention methods). First, each patch of sensors is processed with multi-head temporal self-attention to capture time dynamics; then a graph-based attention is applied across sensors to exchange spatial information. Importantly, STG-Tx employs patchification in the spatial dimension: the 8 600 sensors are partitioned into P=32-sensor groups (patches), and each patch is treated as a “token.” This reduces the number of tokens for spatial attention from 8600 to about 270, drastically cutting the complexity of spatial attention to $O(P^2)$ per head. The patch grouping can be done by geographical proximity or learned embeddings – in our case study, we used a k-nearest neighbor graph to assign sensors to patches such that highly connected sensors fall in the same token. After patchification, STG-Tx applies FlashAttention-2 for temporal attention. FlashAttention is an algorithm that computes exact softmax-attention with reduced memory overhead by tiling and streaming the computations through high-speed on-chip memory. We use a sliding attention window of 256 time steps (approximately 21 hours at 5-minute intervals) to further bound the temporal attention cost. This means each sensor looks at 256 past points within the 288-step (24-hour) look-back window – a slight approximation that ignores the furthest 32 steps, which we found negligible in performance impact. The benefit is that memory usage grows with window size (256) instead of full length (288), and FlashAttention's kernel optimization avoids storing large intermediate activation matrices. Together, patchifying sensors and using FlashAttention-2 enable STG-Tx to perform full quadratic attention (in theory $O(N^2)$ with N large) in practice on the full graph without running out of memory, by effectively limiting N in each domain (time or space) and using GPU memory efficiently.

Hyperparameters and Training Setup: In our largest-case experiment, we set the look-back window to T=288 (covering 24 hours of historical data at 5-min intervals) and the prediction horizon to 12 steps (1 hour ahead). The model dimension is d=256 with 8 attention heads, and L=8 layers (giving a total model size on the order of 5 million parameters). We applied dropout of 0.1 in attention and feed-forward layers to regularize. The model was trained for 50 epochs on 8× NVIDIA A100 80GB GPUs using distributed data parallelism. We used the AdamW optimizer (β_1=0.9, β_2=0.95) with weight decay 10^−2, and a cosine learning rate schedule starting from 10^−3 with 10k warm-up steps. Due to the large batch sizes (we could fit up to 16 historical windows per GPU, which is ~31 million time steps per batch across the cluster), training for 50 epochs (~8 days) was feasible. We also leveraged PyTorch 2.x's torch.compile(..., mode="reduce-overhead") to JIT optimize the model, yielding about an 8 % training throughput gain. Even with full attention on 8600 nodes, our use of FlashAttention and patching meant the peak memory per GPU stayed within 65GB (out of 80GB), avoiding out-of-memory errors. In comparison, a naive space-time attention (as in classical ST-Transformer) on 8600 sensors would be completely infeasible, requiring on the order of 8600^2 ≈ 74 million attention weights per head per layer, and hundreds of GB of memory. By using our efficient design, we preserve the exactness of attention (no loss in theoretical modeling capacity) while cutting memory usage by ~99.8 % relative to such a classical design. This demonstrates how techniques like FlashAttention enable scaling Transformers to unprecedented problem sizes that were previously impractical. The result is a model (STG-Tx) that can be trained on large graphs end-to-end, without resorting to graph down-sampling or extreme model truncation, thus fully leveraging the granular detail of a large traffic network.

Gaps and Positioning

Despite the progress in transformer-based traffic forecasting, several gaps remain in the literature. Firstly, most prior studies validated their models on relatively small-scale sensor networks. Popular datasets like METR-LA and PeMS-Bay have only 207 or 325 sensors, and even the largest PeMS benchmarks (PeMS04, PeMS08) contain on the order of 300–500 nodes. These are tiny compared to real-world traffic networks (California has ~18k sensors). Models that work well on small graphs often face scalability issues on large ones. Indeed, as dataset size grows, the quadratic complexity of naive self-attention and the large number of model parameters lead to prohibitive memory and computation costs. We found that many earlier transformer models did not consider these constraints because the datasets used for evaluation were limited in scale. This leaves a research gap in understanding whether Transformers can maintain their performance when scaled to realistic, large networks. The introduction of the LargeST dataset (8 600 sensors with multi-year coverage) in 2023 directly targets this gap by providing a benchmark for large-scale traffic forecasting. Our review indicates that only very recently have models like STGformer attempted to tackle LargeST; most others have not been tested at this scale.

Secondly, there is a gap in computational efficiency. Transformer models, especially those with many layers or heads, demand heavy compute (GPUs/TPUs) and long training times. This limits their practicality for agencies that may need timely forecasts or to retrain models frequently. For example, STAEformer and PDFormer achieved excellent accuracy but require substantial GPU memory and runtime for training/inference. This motivates the development of simplified transformer variants that retain accuracy while being more efficient. Our STG-Tx is one such attempt, inspired by the successes of STGformer. By simplifying the architecture (fewer layers, factorized attention, etc.), we aim to make deployment on large graphs more feasible. A critical insight from STGformer was that a simpler model can generalize better: it achieved comparable or better accuracy than complex models while using only ~0.2 % of their computation in some cases. This suggests many prior transformer designs might be over-engineered for traffic data, and that distilling the essential parts (global attention + spatial inductive bias) yields a more scalable solution.

Lastly, few works have addressed dynamic and real-time adaptation. Traffic patterns evolve due to seasonality or infrastructure changes. Most transformer models are trained on a fixed historical dataset and tested on a similar domain. But when scaling to a state-wide network and multi-year horizon, one must consider model robustness to distribution shifts (e.g., sensor outages, new congestion patterns). Simpler models are not only faster but often easier to adapt or fine-tune online. This positions our approach (and similar efforts like STGformer) as a foundation for future research into adaptive, large-scale traffic transformers. In summary, the literature to date shows a clear trend: Transformers have advanced the state-of-the-art in traffic forecasting on small and medium networks, but at high computational cost and without full demonstration on large networks. Our work fills this gap by validating a transformer-based model on the full 8 600-sensor California network for the first time, and by proposing architectural simplifications that make such scaling tractable. This contributes a novel perspective that Transformer models can indeed handle large traffic systems when thoughtfully designed. The next section will transition from this literature review to our proposed methodology, where we detail the implementation and evaluation of the STG-Tx model on the California traffic dataset, highlighting how it addresses the gaps identified here.

Summary

In this section, we surveyed transformer-based models for time-series and traffic forecasting. We discussed how self-attention enables learning long-range temporal dependencies more effectively than recurrent or convolutional networks, and reviewed specialized architectures (TFT, Informer, Autoformer) that adapt Transformers to forecasting tasks. We then examined spatio-temporal Transformers for traffic data, noting their strategies to incorporate graph spatial structure (via learned embeddings, delay mechanisms, etc.) and their limitations in scalability. Recent advances such as STGformer demonstrate that with algorithmic optimizations (linear attention) and lean design, Transformers can be scaled to large sensor networks. Our STG-Tx model builds on these ideas, using patch-based dual-path attention and memory-efficient FlashAttention to achieve tractable training on the 8 600-sensor California graph. The insights from this literature review set the stage for the next chapter, where we will introduce our methodology and present experimental results. We aim to show that a carefully engineered Transformer can serve as a practical and accurate solution for large-scale traffic forecasting, bridging the gap between cutting-edge research and real-world deployment.

References

[1] A. Vaswani et al., “Attention Is All You Need,” Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017.
[2] B. Lim et al., “Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting,” Advances in Neural Information Processing Systems, vol. 37, 2021.
[3] H. Zhou et al., “Informer: Beyond Efficient Transformers for Long Sequence Time-Series Forecasting,” Proc. AAAI Conf. Artificial Intelligence, vol. 35, no. 12 2021.
[4] H. Wu et al., “Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting,” Advances in Neural Information Processing Systems, vol. 34, 2021.
[5] H. Liu et al., “STAEformer: Spatio-Temporal Adaptive Embedding Makes Vanilla Transformer SOTA for Traffic Forecasting,” arXiv:2308.10425.
[6] J. Jiang et al., “PDFormer: Propagation Delay-aware Dynamic Long-Range Transformer for Traffic Flow Prediction,” Proc. AAAI Conf. Artificial Intelligence, vol. 37, no. 4, 2023.
[7] H. Wang et al., “STGformer: Efficient Spatiotemporal Graph Transformer for Traffic Forecasting,” arXiv preprint arXiv:2410.00385, 2024.

Chapter 3 The LargeST-CA Dataset

3.1 Dataset Overview

The LargeST-CA dataset represents the largest and most comprehensive publicly available benchmark for traffic forecasting, explicitly designed to evaluate the scalability of spatio-temporal prediction models on highway networks at statewide level [1]. It aggregates five full years of traffic measurements collected from January 2017 through December 2021 by the California freeway monitoring system. In total, the dataset encompasses 8,600 loop detectors, each recording three key traffic variables—flow, speed, and occupancy—at a five-minute sampling frequency. With each sensor contributing 288 time points per day, this results in more than 4.5 billion individual observations across the five-year period.

The dataset was curated and released under the LargeST project, with the explicit objective of establishing a standardized, large-scale testbed for reproducible evaluation of forecasting models [1],[2]. Its design reflects the recognition that prior datasets, though widely used, captured only narrow slices of traffic phenomena, either spatially (focusing on a single metropolitan area) or temporally (covering only a few months). By contrast, LargeST-CA brings together the full heterogeneity of California's freeway system, spanning dense metropolitan areas, suburban corridors, and long rural highways. This breadth ensures that forecasting models trained and evaluated on LargeST-CA are challenged not only by high-dimensional input space but also by the diversity of temporal regimes and topological patterns present across a statewide system.

Unlike earlier benchmarks derived from California's Performance Measurement System (PeMS) that focused on small regional subsets, LargeST-CA provides a unified statewide corpus, enabling the research community to move toward models capable of handling operational-scale networks. This is especially important because real-world ITS deployments are rarely confined to metropolitan subsets; they must operate on entire state or national networks, where coordination across multiple regions becomes essential.

3.2 Sensor Metadata & Spatial Context

Each record in LargeST-CA is associated with a specific inductive loop detector, a sensing technology embedded in freeway pavements that detects passing vehicles by monitoring inductance changes. The dataset enriches these raw measurements with detailed metadata for every sensor. This metadata includes a unique sensor identifier, geographic coordinates (latitude and longitude), and the freeway segment and direction to which the detector belongs. These attributes enable the reconstruction of the road network as a directed graph, where nodes correspond to detectors and edges reflect upstream and downstream connectivity between adjacent freeway segments [2].

The reconstructed spatial graph is large and complex: it consists of approximately 201,000 directed edges, which capture not only simple sequential links but also multiple branching connections at interchanges and merges. This network representation allows forecasting models to move beyond simplistic Euclidean distance assumptions and instead incorporate the true non-Euclidean structure of road traffic. A sensor's flow is not merely correlated with its immediate geographical neighbors; rather, it is influenced by the directed flow of vehicles across connected roadways, including the effects of upstream congestion spilling back into downstream segments.

Importantly, the distribution of sensors across California is highly uneven. Urban areas such as Los Angeles, San Diego, and the San Francisco Bay Area contain dense clusters of detectors, providing detailed coverage of major freeways. In contrast, rural corridors such as Interstate 5 through the Central Valley or long stretches of Interstate 80 in the Sierra Nevada are monitored only sparsely. This spatial imbalance complicates the learning problem. Models must be able to generalize across regions with very different data densities: they must exploit rich information where sensors are abundant while still producing reliable forecasts in areas where detectors are few and far between. Uneven density also raises the challenge of spatial bias, since models may overfit to the dynamics of urban regions while underperforming in rural contexts if not carefully regularized.

The spatial graph thus embodies both an opportunity and a challenge: it provides a rich, non-Euclidean structure that advanced models such as graph neural networks or attention mechanisms can exploit, but it also demands robust generalization strategies that prevent performance disparities between densely and sparsely instrumented regions.

3.3 Temporal Characteristics

A defining strength of LargeST-CA is its continuous coverage across five years, which distinguishes it from nearly all prior benchmarks. This extended temporal horizon provides researchers with unprecedented opportunities to study forecasting under diverse temporal conditions and long-range dependencies. The temporal dimension embodies several layers of variability that forecasting models must address.

At the intra-day scale, traffic exhibits pronounced diurnal rhythms. Morning peaks typically occur between 6:00 and 9:00 AM, corresponding to commuting inflows toward urban centers. Evening peaks are observed between 4:00 and 7:00 PM, often broader in duration due to staggered departure times. Midday volumes stabilize into a plateau, while overnight periods see minimal flow. These cycles are highly consistent in urban corridors but more variable in rural areas, where peak magnitudes are smaller and timing may shift with local conditions.

At the weekly scale, systematic differences arise between weekdays and weekends. Weekdays display strong, predictable commuting peaks, while weekends often exhibit midday peaks associated with leisure and shopping trips. Fridays present a distinctive pattern: the evening peak typically extends later into the night due to early departures and increased recreational travel. Mondays, by contrast, may show a more pronounced morning peak as networks recover from weekend travel.

At the annual scale, traffic volumes fluctuate seasonally. Summer months generally witness reduced commuting demand but increased recreational travel along certain corridors. Holiday periods, such as Thanksgiving and Christmas, produce dramatic changes in traffic flow, with both sharp volume reductions (on the day itself) and extreme congestion immediately before or after. Such calendar-driven seasonality is critical for evaluating model generalization, since it introduces rare but highly impactful patterns.

The dataset also captures exogenous disruptions. Most notably, the onset of the COVID-19 pandemic in 2020 produced abrupt and sustained changes in traffic volumes. Daily peaks flattened, average flows decreased substantially, and recovery proceeded unevenly across regions and time periods. This exogenous shock provides a natural stress-test for forecasting models, since it represents a structural break not observed in training data. A model's ability to adapt to such disruptions is essential for deployment in real-world ITS, where unpredictable events—from pandemics to extreme weather—can significantly alter demand patterns.

The five-minute granularity of the data ensures that both short-lived transients and long-horizon cycles are present. High-frequency sampling allows the dataset to capture rapid dynamics such as shock waves propagating along freeways or sudden drops in speed due to incidents. At the same time, the extended coverage over years incorporates thousands of such events, embedded within seasonal and annual cycles. This multi-scale temporal richness is precisely what makes LargeST-CA a challenging and valuable benchmark for next-generation forecasting models.

3.4 Volume, Sparsity & Data Quality

The scale of LargeST-CA is staggering: with 8,600 sensors each recording 288 observations per day across three variables, the dataset produces roughly 4.5 billion records over its five-year span. This magnitude stresses both storage systems and computational pipelines, requiring models that are not only accurate but also computationally efficient. Handling this data volume motivates the exploration of memory-efficient attention mechanisms, node subsampling strategies, and mixed-precision arithmetic, as discussed in later chapters.

As with all real-world traffic data, LargeST-CA is affected by sparsity and quality issues. Sensors occasionally fail due to hardware malfunctions, communication dropouts, or maintenance activities. Annual missing-value rates of up to five percent per sensor have been reported, with some detectors experiencing extended outages [1]. Beyond missingness, sensors sometimes output erroneous values, such as negative flows or implausibly high occupancies, which must be corrected or filtered during preprocessing.

Unlike curated laboratory datasets that often remove such anomalies, LargeST-CA deliberately preserves these imperfections. The rationale is to ensure that forecasting models are trained and tested under operational conditions, where missing and noisy data are inevitable. This encourages research not only on forecasting per se but also on robust learning strategies that can tolerate incomplete inputs and detect anomalies.

To standardize preprocessing across studies, the dataset provides pre-defined train, validation, and test splits, along with recommendations for Z-score normalization based on global means and standard deviations [2]. By fixing the splits and normalization scheme, the dataset designers ensure that results from different studies remain comparable and reproducible. This is critical for building a cumulative body of knowledge rather than fragmented findings based on inconsistent preprocessing.

In sum, LargeST-CA is not merely a dataset but a carefully constructed benchmarking platform, designed to challenge models with realistic scale and data quality issues while facilitating fair comparison across research efforts.

3.5 Comparison with Prior Public Traffic Sets

Prior to LargeST-CA, the most widely used benchmarks in traffic forecasting included METR-LA (207 sensors, four months), PEMS-BAY (325 sensors, six months), and the PeMSD3/D4/D7/D8 subsets (ranging from 170 to 883 sensors, each covering about two months) [1]. These datasets played a foundational role in the development of STGNNs and early transformer adaptations. For instance, DCRNN [3] and STGCN [4] were primarily evaluated on METR-LA and PEMS-BAY, establishing them as canonical testbeds.

However, these datasets suffer from several limitations. First, their spatial scope is restricted to individual metropolitan areas, which limits the diversity of traffic regimes captured. Second, their temporal coverage is short, typically less than half a year, meaning that models trained on them are never exposed to seasonal cycles or long-term anomalies. Third, their graph sizes are modest, rarely exceeding 1,000 nodes, making them unsuitable proxies for statewide networks. For example, METR-LA includes fewer than 250 detectors in Los Angeles, which is insufficient to capture the complexity of even a single large metropolitan freeway system.

By contrast, LargeST-CA scales up dramatically. The node count increases by more than an order of magnitude (8,600 versus ≤883), the temporal horizon extends to five years, and the dataset integrates traffic from across the entire state. This heterogeneity makes it the first benchmark that truly mirrors the conditions faced by operational ITS. Models that perform well on LargeST-CA are therefore far more likely to scale to real-world deployments.

The dataset's role is thus transformative: it shifts the research paradigm from optimizing architectures for toy-sized graphs toward designing methods that can scale gracefully to operational systems. It also enables evaluation of robustness under structural breaks (such as COVID-19) and across heterogeneous regions, neither of which are represented in prior datasets.

In short, while earlier benchmarks such as METR-LA and PEMS-BAY were invaluable stepping stones, LargeST-CA redefines the landscape of traffic forecasting research. It establishes a new standard by which the scalability, efficiency, and robustness of models can be meaningfully assessed.

Dataset	**# Sensors**	**Duration**	**Sampling**	**Coverage**	**Notes**
METR-LA	207	4 months	5 min	Los Angeles	No seasonal effects
PEMS-BAY	325	6 months	5 min	San Francisco	Medium network size
PeMSD4	307	2 months	5 min	District 4	Small, local subset
PeMSD8	170	2 months	5 min	District 8	Small, rural focus
LargeST-CA	8 600	5 years	5 min	Entire California	Largest scale; long-term dynamics

LargeST-CA thus represents a paradigm shift: models validated only on METR-LA or PEMS-BAY cannot be assumed to scale effectively, since the computational and statistical challenges differ fundamentally. It is the first benchmark where graph size, time horizon, and data quality collectively approximate the demands of real-world statewide deployment.

3.6 Data Preprocessing

Efficient and reproducible preprocessing is an essential prerequisite when scaling learning algorithms to the multi-year and multi-sensor corpus provided by LargeST-CA. Without a carefully designed pipeline, the sheer size of the dataset—billions of records across thousands of detectors—would make training prohibitively slow, memory-intensive, and irreproducible. To address these challenges, this thesis implemented a custom Python preprocessing pipeline that executes the entire workflow in a modular and transparent fashion. The pipeline performs ingestion of raw yearly .h5 files, detection and repair of missing values, construction of supervised temporal windows, normalization of flows, and generation of stable train/validation/test splits. Each stage was engineered to minimize memory footprint, eliminate uncontrolled randomness, and ensure that every experiment can be reproduced exactly, regardless of hardware or runtime environment.

The first stage concerns data ingestion. Raw data are provided as monthly .h5 files, where each column corresponds to a single sensor and each row to a five-minute timestamp. To avoid exhausting system memory, these files are processed sequentially month by month, with intermediate results concatenated into a continuous timeline. During this step, timestamp alignment is explicitly enforced: gaps in the time index are detected, and global sampling frequency is verified to remain exactly five minutes. By working incrementally and using vectorized operations in NumPy whenever possible, the pipeline minimizes intermediate memory copies. When pandas data frames are required, the data are stored in float32 precision rather than float64, reducing memory usage by half without introducing meaningful loss of accuracy in subsequent learning tasks.

The second stage deals with handling missing values. This is critical, as inductive loop detectors frequently exhibit outages or communication dropouts. The implemented strategy operates in two layers. First, a bounded forward–backward fill is applied to bridge short-term gaps, ensuring that brief outages do not break temporal continuity while avoiding artificial smoothing over long interruptions. Using a symmetric fill reduces mean bias compared to a pure forward fill. For longer gaps, the pipeline attempts linear interpolation only when sufficient valid samples exist on both sides; otherwise, missing values are retained and flagged with a mask tensor. This mask is carried forward into training and used in masked loss functions (e.g., masked MAE or RMSE), so the model is not penalized for missing ground truth or inadvertently learning from synthetic imputations. In addition, extreme outliers—such as negative flows or occupancies exceeding physical limits—are clamped or discarded. The aim is to correct calibration errors while preserving genuine peaks that encode important congestion phenomena.

The third stage is construction of supervised learning windows. Forecasting requires input–output pairs, which are generated via a sliding window technique. For each time index t, an input window of 288 steps (corresponding to 24 hours of history) is paired with a prediction target of 12 steps (representing a one-hour horizon). Index validation ensures that both the input and output segments are complete. To accelerate training, two complementary strategies are used: offline storage of windows with a stride greater than one (e.g., storing every 5th window) and dynamic sampling of overlapping windows at runtime using a PyTorch DataLoader. This reduces redundant storage while still exposing the model to the full temporal diversity of the dataset. Optionally, temporal covariates such as time-of-day and day-of-week are added, encoded as continuous variables or sine/cosine pairs to preserve cyclic structure. These features are broadcast across all nodes, enriching the model's ability to learn seasonal and diurnal patterns without greatly increasing dimensionality.

A critical step is normalization. To ensure fair training, a per-sensor Z-score normalization is applied using statistics computed exclusively on the training set. This prevents information leakage into validation and test data. For each sensor, the mean and standard deviation are stored in an auxiliary file, enabling consistent transformation of validation/test samples and exact inversion for evaluation. Per-node normalization is particularly important because flows in dense urban corridors may be orders of magnitude higher than those on rural highways. Without normalization, optimization would be biased toward sensors with larger absolute magnitudes. By contrast, normalization ensures that each node contributes proportionally during training. At evaluation time, inverse normalization is applied so that forecast errors can be reported in natural units (vehicles per five minutes, kilometers per hour, etc.).

The fourth stage addresses the creation of train/validation/test splits. Splits are strictly chronological: 60% for training, 20% for validation, and 20% for testing, with the held-out test period covering 2020–2021. This strategy ensures that the model never trains on future data and mimics real deployment conditions. Random shuffling across the entire timeline was deliberately avoided, since this would yield overly optimistic error estimates by allowing the model to "peek" into future states. Instead, fixed index files (idx_train.npy, idx_val.npy, idx_test.npy) are generated and stored, ensuring that all experiments—regardless of subsequent modifications—use exactly the same examples.

From an implementation perspective, the pipeline was designed to be efficient on both memory and disk. Intermediate results are written into compressed .npz archives, with tensors stored in shape (T, N, C), where T is the number of time steps, N the number of sensors, and C the number of channels/features. Compression not only reduces disk usage but also accelerates subsequent loading during training. Alongside the main data tensor, the pipeline generates a small set of companion files: index arrays for each split, a validity mask for missing values, and a JSON or pickle file containing normalization parameters. This minimalist but complete set of artifacts ensures reproducibility while keeping storage overhead low.

Special emphasis was placed on reproducibility. All sources of randomness that influence data generation, such as the order of window sampling or batch shuffling during training, are controlled by fixed random seeds across NumPy, PyTorch, and CUDA. The pipeline also logs version numbers of all relevant libraries and hashes of the raw input files, making it possible to reconstruct any dataset version exactly. This rigorous control over preprocessing conditions is essential in large-scale machine learning, where even minor differences in data preparation can lead to significant variations in reported performance.

Finally, the pipeline incorporates multiple integrity checks. Before and after each processing stage, summary statistics are computed, including missing-value percentages, ranges and distributions of flow values per node, and total counts of available timestamps. Any anomalies—such as sudden jumps in missingness or implausible ranges—are flagged for inspection. These integrity checks ensure that no silent errors propagate into training, thereby improving trust in the downstream evaluation results.

In sum, the preprocessing pipeline developed for this thesis transforms the raw LargeST-CA corpus into a form that is both computationally tractable and scientifically reliable. By combining efficient handling of massive volumes, robust imputation strategies, strict normalization, and reproducible splits, the pipeline provides a solid foundation upon which scalable forecasting models such as STG-Tx can be trained and evaluated with confidence.

3.6.1 Input Format and Ingestion

The LargeST-CA dataset is released in a modular format, with different types of information distributed across separate files. This organization is deliberate: it keeps raw traffic data, metadata, and network topology distinct, making it easier for researchers to selectively load only what is needed for a given task.

1. Yearly .h5 files (5 total)
    - Each of the five .h5 files corresponds to one calendar year of traffic records, from 2017 through 2021.
    - Within each file, data are stored as a large matrix of shape ($T \times N$), where:
    - T is the number of 5-minute time steps in that year (≈105,120 for a full 365 days).
    - N = 8,600 is the total number of loop detectors.
    - Each column corresponds to a sensor, and each row corresponds to a time step.
    - The stored values represent traffic flow, typically vehicles per 5 minutes.
    - This split avoids creating a single massive file (which would exceed 100 GB in size and be unwieldy to store or download) and makes it easier to parallelize preprocessing across years.
2. Sensor metadata .csv file (1 total)
    - This file contains descriptive information for all 8,600 sensors, including:
    - Sensor ID (unique identifier).
    - Latitude and longitude coordinates.
    - Freeway number and segment information.

- This file is crucial for mapping sensors to geographic space and for reconstructing the spatial graph that underpins spatio-temporal learning.

3. Adjacency .npy file (1 total)
    - This NumPy binary file stores the adjacency matrix of the traffic sensor network.
    - The adjacency is constructed based on road network distances rather than simple Euclidean distance, ensuring that connectivity reflects real traffic flows.
    - Each entry A[i, j] encodes the relationship between sensor i and sensor j, with weights decaying as road distance increases.
    - This file allows researchers to bypass the time-consuming step of constructing a graph from raw geographic coordinates, ensuring reproducibility across experiments.

3.6.2 Temporal Windowing

Transforming Raw Series into Supervised Learning Samples

While the raw traffic time series in LargeST-CA are stored as continuous sequences of sensor readings, machine learning models—particularly deep spatio-temporal architectures—require data to be structured as pairs of inputs and labels. To achieve this, the preprocessing pipeline converts the raw sequences into supervised learning samples using a sliding-window procedure. This method is widely adopted in time-series forecasting because it provides a systematic way to generate overlapping training examples that preserve temporal order while covering the entire dataset.

Input Offsets

For the experiments reported in this thesis, the input window was defined by a look-back length of seq_length_x = 288, corresponding to 24 hours of history at the 5-minute sampling interval. This means that for each supervised sample, the model is given the readings from the previous 288 time steps for all selected sensors. These offsets are represented as integer indices ranging from -287 to 0, where 0 denotes the most recent observation and -287 corresponds to the oldest observation within the look-back horizon.

Encoding offsets in this way allows the model to access not only immediate past dynamics (e.g., congestion buildup over the last 30 minutes) but also long-term periodicity, such as morning and evening peaks or diurnal patterns. By spanning a full day, the input sequence ensures that the model has the contextual information needed to disentangle recurring cycles from transient disturbances.

Output Offsets

The forecast horizon was set to seq_length_y = 12, equivalent to one hour of future traffic evolution at 5-minute resolution. For each training example, the model is tasked with predicting the traffic states at offsets [+1, +2, …, +12] relative to the current time. This setup reflects the operational needs of traffic management systems, which often require forecasts extending up to an hour to inform decisions about signal control, ramp metering, and route guidance.

By structuring labels as multiple future steps rather than a single target, the sliding-window approach supports multi-step forecasting in a direct fashion. This avoids the need to roll forecasts forward recursively (which can compound errors) and instead trains the model to learn the joint distribution of traffic states across an entire hour-long horizon.

Valid Index Computation

Not every time index in the raw dataset can serve as the anchor for a training sample. To construct valid examples, the pipeline computes the range of indices where both sufficient history and sufficient future labels are available. Formally, this range is determined by the combination of minimum and maximum offsets:

- The minimum index required is min(x_offsets) = -287.
- The maximum index required is max(y_offsets) = +12.

Thus, only those time steps that have at least 288 historical values before them and 12 future values after them are eligible for inclusion. For example, the very beginning of the dataset (January 1, 2017, 00:00) cannot produce a valid input because there are not yet 288 preceding time steps. Similarly, the very end of the dataset (December 31, 2021, 23:55) cannot produce a valid output because future labels beyond the dataset boundary are unavailable.

This validity check ensures that every training example is well-formed, containing both complete input histories and corresponding output labels. Once the valid range of indices is determined, the pipeline slides the window across the time axis, extracting supervised samples for each valid position. Because the windows overlap heavily (shifting by one step at a time), the number of training examples generated is very large, providing the model with abundant data for parameter estimation.

Benefits of Sliding Windows

The sliding-window transformation has several key advantages:

1. Consistency with operational forecasting: Each training example mirrors the real-world task of predicting the next hour given the past 24 hours, making the learning objective directly relevant to deployment scenarios.
2. Exploitation of temporal redundancy: Overlapping windows allow the model to repeatedly see traffic states in slightly different contexts, improving robustness and generalization.
3. Flexibility: The choice of input and output lengths (seq_length_x and seq_length_y) can be adjusted depending on the application, whether it prioritizes short-term responsiveness or longer-term planning.

3.6.3 Feature Engineering

In addition to the raw traffic flow values recorded by loop detectors, the preprocessing pipeline incorporates temporal context features designed to encode the periodic structures inherent in traffic demand. These features provide the model with auxiliary information about the position of each observation in the daily and weekly cycles, allowing it to better capture recurrent patterns such as morning peaks, evening rush hours, and weekend effects. Without such contextual cues, the model would need to infer periodicity implicitly from sequences of flow values, a task that is both less efficient and more prone to overfitting.

Time-of-Day (ToD)

The first contextual feature is Time-of-Day (ToD), which encodes the precise position of each timestamp within a 24-hour cycle. ToD is computed as a fractional value in the range [0,1):

$$\text{ToD}(t) = \text{minutes since midnight} / 1440$$

where 1440 represents the number of minutes in a day. For example, a timestamp at 6:00 a.m. (360 minutes after midnight) would yield ToD = 0.25, while a timestamp at 6:00 p.m. (1080 minutes after midnight) would yield ToD = 0.75.

This encoding normalizes the daily cycle and provides the model with a continuous representation of diurnal position. By feeding ToD as a feature alongside flow values, the model can more easily associate particular times of day with expected traffic behaviors. Morning rush hours, for example, occur reliably around ToD ≈ 0.25–0.33, while evening congestion tends to cluster near ToD ≈ 0.70–0.80. Explicitly providing this feature therefore reduces the burden on the model to infer periodicity from lagged observations alone.

Day-of-Week (DoW)

The second contextual feature is Day-of-Week (DoW), which encodes weekly periodicity. Each timestamp is assigned a normalized weekday index, where 0 corresponds to Monday and 6 corresponds to Sunday. This index is then normalized to a fractional range (e.g., dividing by 7) so that values lie between 0 and 1.

The DoW feature is essential for distinguishing between weekday and weekend traffic dynamics. For instance, commuting flows are prominent Monday through Friday but attenuate on Saturdays and Sundays, while recreational travel may peak disproportionately on weekends or holidays. Incorporating DoW as a feature allows the model to recognize such weekly cycles and adjust its predictions accordingly.

Representation Across Sensors

Both ToD and DoW features are tiled across all sensors, ensuring that each detector receives the same contextual information for a given timestamp. This results in a feature tensor of shape (T, N, C)

- T is the number of timesteps in the dataset (≈525,600 for five years at 5-minute intervals),
- N = 8,600 is the number of loop detectors, and
- C is the number of channels. In this configuration, C = 3: raw flow values, ToD, and DoW.

This design means that for every observation in the dataset, the model has simultaneous access to both the traffic flow measurement and its temporal context, expressed in a standardized numerical form.

Benefits for Forecasting

The addition of ToD and DoW channels provides multiple benefits:

1. Improved learning efficiency. By explicitly encoding cyclical information, the model can allocate capacity to learning irregularities and disruptions rather than rediscovering obvious periodicity.

2. Better generalization. Context features make the model less sensitive to missing or corrupted flow values, since the temporal position can still guide expectations.

3. Alignment with real-world operations. Traffic management systems often operate on schedules defined by time-of-day and day-of-week (e.g., peak-hour signal plans). Including these features makes the model's predictions more interpretable and compatible with existing ITS frameworks.

3.6.4 Dataset Partitioning

Once supervised samples are generated, they are partitioned into chronological splits (e.g., 60% training, 20% validation, 20% testing, as described in Chapter 4). Importantly, the sliding-window approach respects temporal ordering: training samples always precede validation samples, which in turn precede test samples. This prevents information leakage across splits and ensures that evaluation reflects genuine out-of-sample generalization.

3.6.5 Normalisation

To stabilise optimisation, flow values are Z-score normalised per sensor:

$$\tilde{x} = \frac{x - \mu}{\sigma},$$

where mean μ and standard deviation σ are estimated only from the training slice (to avoid test leakage). A lightweight StandardScaler class stores these statistics and provides both forward and inverse transforms for evaluation.

3.6.6 Output Artifacts

The processed dataset is written to a dedicated subdirectory (<dataset>/<years>/) with the following artifacts:

- his.npz – compressed NumPy archive containing the full tensor (T, N, C) and the normalisation statistics (mean, std).
- idx_train.npy, idx_val.npy, idx_test.npy – index arrays defining the splits.

Together, these files provide a memory-efficient and portable format for downstream PyTorch training.

Summary

The preprocessing pipeline converts ~4.5 billion raw records into model-ready tensors through (i) ingestion and NaN repair, (ii) temporal sliding-window construction, (iii) optional ToD/DoW feature augmentation, (iv) chronological splitting, and (v) per-sensor Z-score normalisation. The entire workflow executes in a single pass, producing reproducible artefacts that can be shared across experiments. This standardisation ensures comparability with prior work on LargeST-CA and removes data-handling as a bottleneck in large-scale Transformer training.

References

[1] X. Liu, Z. Qin, Y. Zhao, H. Wang, and Y. Li, “LargeST: A Benchmark Dataset for Large-Scale Traffic Forecasting,” NeurIPS Datasets and Benchmarks Track, 2023, arXiv:2306.08259.
[2] X. Liu, “LargeST Benchmark Dataset (GitHub Repository),” https://github.com/liuxu77/LargeST, accessed 14 Jun 2024.

3.7 Model Architecture: STG-Tx (Simplified Spatio-Temporal Graph Transformer)

The model developed in this thesis, termed STG-Tx, is a simplified yet scalable spatio-temporal transformer tailored for traffic forecasting at state-wide scale. Its design explicitly addresses the dual challenge of modeling temporal dynamics of traffic flows and the spatial interactions among thousands of road sensors. Unlike classical recurrent or convolutional architectures, the transformer framework naturally supports long-range dependencies through self-attention [1], making it a suitable foundation for large-scale traffic prediction.

Temporal Component

The temporal pathway of STG-Tx processes sequences of past observations for each traffic sensor. Each input is projected into a latent dimension through a linear embedding layer, followed by sinusoidal positional encoding [1] to retain the order of time steps. This design allows the model to capture both short-term fluctuations (such as rush-hour spikes) and longer seasonal dependencies.

Spatial Component

Spatial dependencies are represented through node embeddings that act as learnable identifiers for each of the 8 600 sensors. These embeddings are broadcast across time steps and added to the temporal representations, enabling the model to encode structural and topological information about the sensor network. Unlike more complex graph convolutional models such as DCRNN [2] or STGCN [3], STG-Tx opts for this simplified embedding strategy to remain computationally feasible at scale.

Transformer Encoder

The core architecture leverages a multi-layer transformer encoder [1], composed of stacked self-attention and feed-forward blocks. Each encoder layer uses multi-head self-attention to jointly model interactions across time and sensors, with GELU activation, dropout regularization, and pre-layer normalization. Importantly, the encoder is implemented with memory-efficient PyTorch primitives [4], ensuring that even sequences of length 288 (24 hours at 5-minute intervals) can be processed without exceeding GPU limits.

Readout and Output

The encoder's output is reduced by selecting the final time-step representation, which is then processed by a prediction head: a small multi-layer perceptron that generates multi-step forecasts

(12 steps, i.e., 1 hour ahead). The model is trained using a masked mean absolute error (MAE) loss, and predictions are produced for all sensors simultaneously.

Efficiency Considerations

Efficiency was a core design requirement. To make training on the LargeST-CA dataset [5] feasible, STG-Tx introduces two strategies:

- Node subsampling during training, which selects a random subset of sensors per batch, reducing memory load while maintaining diversity across epochs.
- Mixed-precision training using PyTorch's AMP [6], which accelerates computation and reduces GPU memory consumption by operating in FP16 when safe.

This combination of architectural simplification and engineering optimization allows STG-Tx to scale to the full California network of 8 600 sensors, something that earlier spatio-temporal models struggled to achieve [2],[3].

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention Is All You Need,” in Advances in Neural Information Processing Systems (NeurIPS), 2017.
[2] Y. Li, R. Yu, C. Shahabi, and Y. Liu, “Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting,” arXiv:1707.01926
3] B. Yu, H. Yin, and Z. Zhu, “Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting,” arXiv:1709.04875
[4] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., “PyTorch: An Imperative Style, High-Performance Deep Learning Library,”arXiv:1912.01703
[5] X. Liu, Z. Qin, Y. Zhao, H. Wang, and Y. Li, “LargeST: A Benchmark Dataset for Large-Scale Traffic Forecasting,” arXiv:2306.08259
[6] M. Micikevicius, P. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, and H. Wu, “Mixed Precision Training,” arXiv:1710.03740

Chapter 4 - Hardware and Environment

The experimental setup is a crucial element in any large-scale machine learning study, especially when the research problem involves processing vast amounts of spatio-temporal traffic data across thousands of sensors and multiple years. For this thesis, all experiments were performed on a cloud-hosted environment provided by Paperspace, leveraging a single NVIDIA A100 GPU equipped with 80 GB of VRAM, supported by 96 virtual CPUs and 512 GB of system RAM. This configuration was selected after extensive consideration of both the memory footprint required to handle the LargeST-CA dataset and the computational intensity of training deep spatio-temporal transformer models. The A100 GPU represents one of the most powerful accelerators available for large-scale deep learning workloads, offering not only sheer memory capacity but also architectural optimizations for tensor operations that align closely with transformer-style models [1].

The choice of GPU hardware is particularly relevant in this context because traffic forecasting models, when scaled to thousands of sensors, must handle extremely large adjacency matrices, long temporal sequences, and high-dimensional embeddings. For example, the STG-Tx model used in this study employs a model dimension (d_model) of 256 and processes 288 timesteps per sequence, with learnable embeddings for all 8,600 sensors in the California network. Without sufficient GPU memory, it would be impossible to hold the intermediate activations of such a model in memory during training, let alone process a batch of multiple sequences in parallel. The A100's 80 GB VRAM was essential to train the Variant C (Large) of STG-Tx, which otherwise would require partitioning the dataset or shrinking the model, both of which could compromise the fidelity of the results.

The computational environment was standardized to ensure reproducibility and determinism. All scripts were executed under Ubuntu 22.04 LTS, a long-term support version of the Linux operating system that provides stability for scientific software environments. The machine learning framework employed was PyTorch 2.2 [4], compiled with CUDA 12.0 support, ensuring compatibility with the A100's Ampere architecture and optimized tensor cores. Python 3.11 was chosen for its performance improvements and compatibility with modern machine learning libraries. Random seeds were explicitly fixed across NumPy, PyTorch, and CUDA, which is essential to guarantee reproducibility of results across multiple runs. Without such controls, small variations in initialization or execution order could propagate into differences in model training, making it difficult to draw consistent conclusions. Determinism is particularly critical in the context of spatio-temporal learning, where subtle changes in initial conditions can lead to divergent training outcomes.

By carefully specifying the hardware and software environment, this study ensured that the experimental results were not artifacts of hardware constraints or software inconsistencies, but rather reliable reflections of the underlying model design and training protocol. The environment was therefore not just a technical detail, but a fundamental enabler of the research's scalability and validity.

4.2 Dataset Splits

The dataset used in this thesis was LargeST-CA, which spans five years of data (2017–2021) and covers the entire state of California with 8,600 traffic sensors [5]. Its unprecedented size and coverage make it the largest publicly available dataset for traffic forecasting and therefore a natural choice for evaluating the scalability of spatio-temporal transformer models. However, its scale also necessitated careful decisions in preprocessing and data splitting to ensure that training, validation, and test sets reflected real-world forecasting scenarios.

The inputs to the model comprised 288 timesteps, representing a full 24-hour history at 5-minute intervals. This length was selected to provide the model with one complete daily cycle of traffic patterns, capturing both the morning and evening peaks as well as the off-peak periods. Forecast horizons were set to 12 timesteps, equivalent to one hour ahead, which is a practical horizon for operational traffic forecasting – long enough to be useful for control decisions, but short enough to retain high predictive accuracy.

Data splitting was conducted chronologically, with 60% of the data assigned to training, 20% to validation, and 20% to testing. Importantly, the test split was drawn from the held-out period of 2020–2021, ensuring that evaluation was performed on completely unseen temporal data. This chronological splitting strategy is critical for avoiding information leakage, which could occur if training and testing samples were randomly interleaved. Random splitting is unsuitable in traffic forecasting because it risks training the model on data from the same temporal period as the test set, artificially inflating performance. Chronological splits ensure that the model is tested on genuinely unseen time horizons, simulating the real-world task of predicting future traffic conditions from past data.

The choice of validation set as 2019 data provided a robust mechanism for tuning hyperparameters while still respecting the chronological order. Validation data thus acted as a “proxy future,” giving feedback on how well the model generalizes to unseen periods before final testing. Such splitting is consistent with best practices in time-series forecasting, where temporal continuity must be respected.

Furthermore, this splitting strategy allowed the evaluation of model robustness across different traffic regimes, including pre-pandemic and pandemic conditions. Traffic in 2020–2021 was heavily disrupted by COVID-19, leading to reduced peak demand, altered commuting patterns, and higher volatility. Evaluating on this period thus provided a stringent test of the model’s generalization capacity. A model that performs well both in stable pre-pandemic years and in the volatile pandemic years can be considered robust and adaptable to changing traffic conditions, an important trait for real-world deployments.

4.3 Model Configuration

The model configuration employed in this study was specifically designed for statewide forecasting. This configuration significantly scales up the representational capacity of the model compared to smaller variants, with the following parameters: a model dimension of 256, a feed-forward dimension of 2048, 8 attention heads, 8 transformer layers, and a dropout rate of 0.1. Each of these hyperparameters was chosen based on theoretical considerations and empirical evidence from both the transformer literature and prior traffic forecasting studies.

The model dimension (256) determines the size of the embeddings and the internal representation of the network. A larger dimension increases the expressive power of the model, allowing it to capture more subtle patterns in the traffic data. This choice was informed by prior transformer work [1], which demonstrated that increasing model dimension leads to substantial performance improvements up to a point, after which returns diminish. For the LargeST-CA dataset, with its vast number of sensors and complex spatio-temporal patterns, a smaller dimension (e.g., 128 or 256) proved insufficient in preliminary experiments, leading to underfitting. Conversely, dimensions larger than 512 were found to exceed memory capacity without meaningful accuracy gains.

The feed-forward dimension (2048) refers to the size of the intermediate layer in the transformer's position-wise feed-forward networks. This expansion ratio of 4× relative to the model dimension is standard practice in transformer design [1] and allows the model to capture non-linear interactions more effectively. The 8 attention heads permit the model to jointly attend to information from multiple representation subspaces. In traffic forecasting, this means the model can simultaneously consider different aspects of spatio-temporal dependencies – for example, one head might specialize in capturing daily periodicity, while another focuses on short-term fluctuations during incidents.

The 8 layers in the encoder stack enable deep hierarchical representation learning, with each successive layer refining and abstracting the input features. Deeper models generally achieve higher accuracy, but at the cost of increased training time and memory usage. The choice of 8 layers represents a balance between depth and efficiency, sufficient to capture complex interactions across space and time without overwhelming computational resources.

Dropout with a rate of 0.1 was employed as a regularization mechanism to prevent overfitting. Despite the large size of the dataset, overfitting remains a risk due to the high capacity of transformer models. Dropout helps ensure that the model generalizes to unseen periods rather than memorizing specific traffic patterns from the training data.

The model also incorporated learnable per-node embeddings for all 8,600 sensors. These embeddings provide each sensor with a unique identifier in the latent space, enabling the model to differentiate between locations while still learning shared patterns across the network. Unlike fixed adjacency matrices used in graph convolutional networks [2],[3], this approach allows the model to learn spatial relationships dynamically from the data.

Two specialized mechanisms were introduced to enhance scalability. First, the temporal encoder operated over the full 288-step input using FlashAttention, an algorithm designed for memory-efficient self-attention. FlashAttention reduces the quadratic memory and compute complexity of standard attention, enabling the model to process long input sequences without exceeding GPU capacity. Second, the spatial encoder employed attention across node patches with subsampling. Rather than attending over all 8,600 nodes simultaneously – which would be prohibitively expensive – the model processes subsets of nodes (patches), sampled in a structured way to preserve spatial coherence. This patch-based approach maintains global coverage while reducing per-step computation, a design choice directly inspired by scalability concerns.

Together, these architectural choices allowed STG-Tx (Large) to scale up to the statewide forecasting task while preserving efficiency through FlashAttention and patchified node sampling. The configuration demonstrates how modern transformer techniques can be adapted to the unique demands of large-scale spatio-temporal traffic forecasting.

4.4 Training Protocol

The training protocol was carefully designed to balance stability, efficiency, and convergence speed. Training was conducted for 50 epochs using the AdamW optimizer [7], with a base learning rate of 1×10^{-4}. AdamW was chosen due to its decoupled weight decay mechanism, which has been shown to improve generalization and stability compared to standard Adam. In large-scale transformer training, AdamW is now considered the default optimizer due to its robustness across diverse tasks.

The optimizer hyperparameters were set to $\beta_1 = 0.9$ and $\beta_2 = 0.999$, reflecting standard values that provide stable adaptive moment estimates. A weight decay of 1×10^{-4} was applied to regularize the network and reduce overfitting. Without weight decay, the model risked memorizing idiosyncrasies of the training data, especially given the high capacity of the 8-layer, 256-dimension transformer.

To further stabilize training, gradient clipping at 1.0 was employed. Gradient clipping prevents exploding gradients, which can occur when training deep networks on long sequences. This measure ensures that updates remain controlled and prevents destabilization of the optimization process.

A cosine annealing schedule with 5,000 warm-up steps was used to adapt the learning rate over time. During the warm-up phase, the learning rate gradually increased, allowing the optimizer to escape poor local minima and achieve stable convergence. After warm-up, the cosine decay ensured smooth reduction of the learning rate, promoting fine-tuning in later epochs. This schedule has become standard in transformer training, as it balances exploration and exploitation effectively.

The batch configuration was tailored to maximize GPU utilization while respecting memory constraints. Each batch comprised 8 sequences per GPU, with 1,024 nodes per batch during training. For evaluation and testing, 4,096 nodes per chunk were processed, ensuring coverage of the entire network without exceeding GPU capacity. This chunking approach reflects a practical compromise: while training can subsample nodes to reduce computational load, evaluation must eventually generate predictions for all sensors. The use of node chunking allowed evaluation on the full dataset in manageable portions.

Automatic Mixed Precision (AMP) [6] was enabled throughout training to enhance efficiency. AMP leverages half-precision floating point (FP16) for most operations while retaining full precision (FP32) for critical accumulations, reducing memory usage and accelerating training without significant loss of accuracy. The use of AMP was especially beneficial in this study given the scale of the model and dataset. Without AMP, it would have been infeasible to train the STG-Tx Large variant within the memory budget of the A100.

Checkpoints were saved regularly, and the best checkpoint was selected based on validation MAE. This ensured that the model retained the weights corresponding to the lowest error on unseen data, rather than simply the final epoch. Early stopping with patience of 8 epochs was also applied, preventing unnecessary training once convergence had been reached. Early stopping not only saves computation but also reduces the risk of overfitting.

This training protocol, incorporating AdamW optimization, gradient clipping, learning rate scheduling, AMP, and robust checkpointing, reflects the best practices in modern deep learning. Each design choice was motivated by the need to ensure stable convergence, efficient use of hardware, and reproducible outcomes.

4.5 Monitoring and Logging

Monitoring and logging are critical components of large-scale experiments, ensuring both transparency of results and the ability to diagnose issues during training. In this study, training progress was logged every 25 iterations, recording both batch-level MAE and RMSE. These metrics provided immediate feedback on the model's performance, highlighting trends such as rapid convergence, plateauing, or divergence.

Validation was conducted at the end of each epoch, with the same metrics computed on the validation set. This practice allowed for continuous monitoring of generalization performance, preventing the model from overfitting to the training data without detection. The use of both MAE and RMSE provided complementary perspectives on performance. MAE captures the average absolute error, which is robust to outliers and easy to interpret in terms of units of traffic flow or speed. RMSE, by squaring errors, penalizes large deviations more heavily, offering sensitivity to worst-case scenarios. Together, these metrics allowed for balanced evaluation.

At the end of training, the best validation checkpoint was evaluated on the held-out test set (2020–2021) using MAE, RMSE, and MAPE. Including MAPE in the final evaluation added interpretability by expressing errors as percentages relative to true values, although care was taken to interpret this cautiously given that low-flow periods can inflate percentage errors. By triangulating across multiple error measures, the evaluation provided a more comprehensive picture of performance than any single metric alone.

All logs, checkpoints, and evaluation results were archived systematically. This archival process not only facilitated analysis and reporting but also ensured that experiments could be revisited or extended in the future without re-running the entire pipeline.

4.6 Reproducibility

Ensuring reproducibility is a cornerstone of credible scientific research, particularly in the field of machine learning where results can be sensitive to seemingly minor variations in setup. In this thesis, reproducibility was prioritized at every stage of the experimental workflow.

All preprocessing, training, and evaluation scripts were placed under version control, with changes tracked in a dedicated repository. This practice ensured that the exact code corresponding to reported results could be retrieved and inspected. Version control also facilitated collaboration and safeguarded against accidental loss of experimental configurations.

Datasets were stored in compressed .npz format, which balances efficiency in storage with rapid loading during training. Fixed splits were saved as explicit index files (idx_train.npy, idx_val.npy, idx_test.npy), ensuring that the same training, validation, and test sets were used consistently across experiments. This eliminated ambiguity and prevented accidental data leakage.

Determinism was enforced by fixing random seeds across NumPy, PyTorch, and CUDA. This step is often overlooked in deep learning research, but it is essential when results must be reproducible. Without fixed seeds, even the same code run twice on the same hardware can yield slightly different results due to differences in initialization or parallel thread scheduling. By controlling these factors, experiments in this study produced consistent outcomes across repeated runs.

Finally, all experiments were executed within a single Paperspace workspace, ensuring consistency in hardware and environment. This avoided the pitfalls of running some experiments on different machines, which could introduce confounding factors such as different GPU architectures or software versions.

By adhering to these practices, this study provides not just results but also a reproducible methodology that can serve as a baseline for future work in large-scale traffic forecasting. Other researchers can replicate the results by following the documented environment setup, dataset preparation, and training protocol, thereby advancing transparency and trust in the research community.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention Is All You Need,” in Advances in Neural Information Processing Systems (NeurIPS), 2017. [Online]. Available: https://arxiv.org/abs/1706.03762
[2] Y. Li, R. Yu, C. Shahabi, and Y. Liu, “Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting,” in International Conference on Learning Representations (ICLR), 2018. [Online]. Available: https://arxiv.org/abs/1707.01926
[3] B. Yu, H. Yin, and Z. Zhu, “Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting,” in International Joint Conference on Artificial Intelligence (IJCAI), 2018. [Online]. Available: https://arxiv.org/abs/1709.04875
[4] A. Paszke et al., “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” in Advances in Neural Information Processing Systems (NeurIPS), 2019, arxiv:1912.01703
[5] X. Liu, Z. Qin, Y. Zhao, H. Wang, and Y. Li, “LargeST: A Benchmark Dataset for Large-Scale Traffic Forecasting,” arXiv:2306.08259
[6] M. Micikevicius et al., “Mixed Precision Training,” in International Conference on Learning Representations (ICLR), 2018, arxiv:1710.03740
[7] I. Loshchilov and F. Hutter, “Decoupled Weight Decay Regularization,” in International Conference on Learning Representations (ICLR), 2019, arxiv:1711.05101

Chapter 5 – Experimental Results

This chapter presents the empirical findings of the thesis, organised into five sections: the training and validation dynamics of the STG-Tx model, its final performance on the held-out test set, a comparison against baseline spatio-temporal architectures, an ablation study of efficiency techniques, and a discussion of scalability on the LargeST-CA benchmark.

5.1 Training and Validation Behavior

Training of the STG-Tx architecture was conducted for a total of 50 epochs on the LargeST-CA dataset, following the experimental configuration outlined previously in Chapter 4. This training schedule was chosen to balance computational feasibility with the need to allow the model sufficient opportunity to converge. Each epoch encompassed several thousand gradient updates, given the size of the dataset and the design of the temporal input windows. The model was trained end-to-end, with both the temporal and spatial attention modules updated simultaneously.

To ensure that the model could be trained efficiently at statewide scale, the batch size was fixed at 8 sequences per GPU. Each sequence corresponded to a 24-hour historical window (288 time steps) across a selected subset of nodes. This batch size represented a practical compromise between GPU memory limits and the statistical stability of gradient estimates. Smaller batch sizes would have increased gradient noise and slowed convergence, while larger batches were infeasible given the memory requirements of storing high-dimensional temporal-spatial embeddings.

Because of the unprecedented scale of the dataset, node subsampling was employed as a core efficiency strategy. At each iteration, rather than processing all 8,600 sensors simultaneously, a random subset of nodes was selected for inclusion in the mini-batch. This reduced the effective dimensionality of the attention mechanisms, keeping the model within the memory and compute budgets of a single NVIDIA A100 GPU. While this introduced a degree of stochasticity into training, it allowed the network to gradually learn from the full sensor graph over multiple epochs. Over the course of training, the sampling procedure ensured that all regions of the network were represented, albeit not in every batch, creating a balance between efficiency and representational coverage.

The optimization process was further stabilized through the combined use of gradient clipping and mixed-precision arithmetic (AMP). Gradient clipping was set to a maximum norm of 1.0, which prevented occasional spikes in gradient magnitudes from leading to divergence or catastrophic updates. This measure was particularly important during the early epochs, when weights are uncalibrated and large gradients are most likely. Meanwhile, AMP enabled the training process to alternate between 16-bit and 32-bit precision operations. This reduced memory usage and increased throughput without degrading numerical stability. Together, these techniques ensured that training proceeded smoothly, with no evidence of exploding gradients (where updates grow uncontrollably large) or vanishing gradients (where updates shrink toward zero). The absence of such instability is notable, since both phenomena are common in deep sequence models, especially when trained on long input windows such as the 288-step histories used here.

Monitoring of the validation curve provided insight into convergence behavior. Across multiple runs, validation mean absolute error (MAE) decreased steadily for the first 30 epochs, reflecting the model's increasing ability to capture both temporal and spatial dependencies in the data. Between epochs 30 and 38, improvements became progressively smaller, suggesting that the model was approaching its optimal representational capacity under the current configuration. After epoch 38, the validation curve began to plateau, with only marginal gains despite continued training. This flattening indicated that further epochs were unlikely to yield substantial benefits and that the model had effectively saturated the predictive signal available from the dataset given its architectural constraints.

The lowest validation MAE was consistently achieved around epoch 38, making this checkpoint the natural choice for model selection. To avoid overfitting, early stopping was implemented with a patience of eight epochs, ensuring that the final chosen model reflected the best generalization performance rather than the lowest training error. The checkpoint at epoch 38 was therefore designated as the final model for evaluation on the held-out test set (covering 2020–2021). This choice reflects standard practice in deep learning, where the model corresponding to the best validation score, rather than the final epoch, is retained for downstream evaluation.

In summary, the training process confirmed the stability and efficiency of the STG-Tx architecture. Despite the massive scale of the dataset, the combination of moderate batch size, node subsampling, gradient clipping, and AMP enabled the model to converge reliably within 40 epochs, without requiring specialized hardware beyond a single high-memory GPU. The observed convergence dynamics further validate the design choices of the architecture: the model was sufficiently expressive to capture the dominant predictive signals but not so complex as to overfit or require excessive training time.

5.2 Test Set Performance

Following the identification of the best-performing checkpoint at epoch 38, the model was evaluated on the held-out test split covering the years 2020–2021. This test period was deliberately chosen to represent unseen temporal regimes, including the significant distribution shift associated with the COVID-19 pandemic, in order to provide a robust measure of generalization beyond the training interval. Performance was quantified using three widely adopted error metrics in the traffic forecasting literature: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE).

The first metric, MAE, captures the average magnitude of deviations between predicted and observed traffic variables, expressed in natural units (e.g., vehicles per five minutes or kilometers per hour depending on the channel). On the test set, STG-Tx achieved an MAE of 30.2, indicating that on average, predictions deviated by about 30 vehicles per five-minute interval. This level of error is considered competitive for statewide forecasting, given the sheer scale of the dataset and the heterogeneity across 8,600 sensors. Importantly, MAE is insensitive to the direction of errors—

whether the model overpredicts or underpredicts—and therefore reflects the baseline magnitude of prediction inaccuracies without penalizing large outliers disproportionately.

The second metric, RMSE, amplifies the impact of larger deviations by squaring the residuals before averaging and then taking the square root. STG-Tx attained an RMSE of 39.7, which, while higher than the MAE as expected, remained within a modest range. The relatively small gap between MAE and RMSE suggests that while occasional large errors occurred, they did not dominate overall performance. This is significant because in traffic forecasting, catastrophic failures—such as completely missing the onset of a major congestion event—can undermine the operational utility of a model. The results indicate that STG-Tx avoids such failures with reasonable consistency, maintaining reliability across both typical and atypical conditions.

The third metric, MAPE, expresses error as a percentage of the observed values, providing a scale-independent measure. STG-Tx recorded a MAPE of 8.3%, meaning that the model's forecasts were, on average, within about 8% of actual observed traffic volumes. This figure is particularly useful for comparing performance across sensors with widely varying flow magnitudes. For instance, a deviation of 30 vehicles is far more consequential at a lightly traveled rural site than at a heavily congested urban interchange. By normalizing error relative to the magnitude of observed values, MAPE highlights the model's capacity to generalize across regions with different traffic intensities. Achieving a single-digit MAPE across such a heterogeneous statewide network underscores the robustness of STG-Tx.

From a practical standpoint, these error levels suggest that STG-Tx is sufficiently accurate for many short-term ITS applications. An MAE of 30 vehicles per interval translates to errors of only a few percent relative to the capacity of a typical freeway lane, which is unlikely to materially degrade control decisions such as ramp metering rates or dynamic message sign updates. Likewise, a MAPE below 10% ensures that route guidance systems can provide drivers with reasonably accurate travel-time predictions, even in the presence of noisy or incomplete sensor inputs. While further refinements may be needed for applications demanding ultra-precise forecasts (e.g., congestion pricing strategies), the results provide strong evidence that transformer-based architectures can be deployed effectively for operational traffic management at statewide scale.

In summary, the test set evaluation demonstrates that STG-Tx achieves a compelling combination of accuracy, robustness, and scalability. The trio of metrics—MAE 30.2, RMSE 39.7, and MAPE 8.3%—provides convergent evidence of strong predictive power across diverse horizons and sensor contexts. By validating on a held-out period that included unprecedented disruptions, the evaluation also underscores the model's resilience under real-world conditions. These findings reinforce the central claim of this thesis: transformer-based models, when carefully adapted, can deliver state-level traffic forecasts that are both accurate and operationally viable.

5.3 Comparison with Baseline Architectures

To contextualize the performance of STG-Tx, it is necessary to situate the results relative to established baseline architectures in the traffic forecasting literature. While STG-Tx introduces a novel transformer-based framework with efficiency-oriented design, its contributions can only be fully appreciated when compared against representative models spanning the main evolutionary stages of spatio-temporal forecasting. Accordingly, five well-known baselines were selected: the Diffusion Convolutional Recurrent Neural Network (DCRNN) [1], the Spatio-Temporal Graph Convolutional Network (STGCN) [2], the Long Short-Term Memory (LSTM) network, the Graph WaveNet (GWNET), and the Spatio-Temporal Graph Neural Ordinary Differential Equation (STGODE) model. These architectures collectively represent the progression from classical recurrent models, through graph convolutional hybrids, to more sophisticated spatio-temporal designs.

Overview of Baseline Models

The DCRNN [1] was one of the earliest models to explicitly combine graph-based spatial learning with recurrent temporal modeling. It leverages diffusion convolution to capture directed spatial dependencies on traffic networks, coupled with gated recurrent units (GRUs) to model temporal evolution. DCRNN remains a strong baseline on smaller benchmarks (e.g., METR-LA, PEMS-BAY), but its reliance on sequential recurrence at every time step makes it computationally expensive when scaled to statewide datasets such as LargeST-CA. Training requires multiple passes of graph convolution per time step, and with 288-step input sequences, the computational burden becomes prohibitive.

The STGCN [2] represents a subsequent generation of spatio-temporal models that integrates temporal convolutional filters with graph convolutional layers. By replacing recurrence with temporal convolutions, STGCN reduces sequential dependencies and accelerates training. However, like DCRNN, it was primarily evaluated on relatively small urban datasets and has not demonstrated scalability to graphs on the order of thousands of nodes. Furthermore, temporal convolutions, while efficient, may struggle to capture very long-range dependencies compared to attention-based mechanisms.

The LSTM baseline provides a useful point of reference as a purely temporal model without explicit spatial structure. Long Short-Term Memory networks have been widely applied to traffic forecasting due to their ability to capture nonlinear sequence dependencies and their robustness across diverse datasets. However, because LSTM architectures treat each sensor independently, they cannot exploit the rich spatial correlations inherent in road networks. As such, while LSTM models provide competitive accuracy on short-term, single-link forecasts, they typically underperform in network-wide settings where spatial spillovers are critical.

Graph WaveNet (GWNET) represents a more advanced architecture that combines dilated temporal convolutions with adaptive graph learning. By learning a dynamic adjacency matrix rather than relying solely on fixed graph structures, GWNET introduced greater flexibility in modeling non-Euclidean dependencies. This adaptability made it a strong performer on mid-scale datasets, but the cost of computing adaptive adjacency matrices increases sharply with node count, presenting challenges at the scale of LargeST-CA.

Finally, STGODE integrates the framework of neural ordinary differential equations into the spatio-temporal setting. By parameterizing traffic evolution as a continuous-time dynamic system, STGODE offers theoretical elegance and strong performance on smaller datasets. However, the method is computationally demanding, requiring iterative ODE solvers during training and inference, and thus faces similar scalability bottlenecks when applied to very large graphs with long temporal horizons.

Comparative Performance and Limitations

Due to the computational demands of reproducing full-scale training for all baselines, this study relied on published benchmark results where available, complemented by limited in-house experiments with reduced graph sizes. Table 5.2 reports performance figures side by side, drawing from authoritative sources in the literature. While the absolute numbers vary depending on dataset and experimental setup, certain patterns emerge consistently.

First, LSTM-based models, while serviceable for short sequences, consistently trail behind graph-based methods when evaluated on network-wide forecasting tasks. Their inability to exploit spatial correlations renders them less effective in capturing congestion propagation, leading to higher errors across all metrics.

Second, DCRNN and STGCN remain competitive on smaller datasets, but their training costs scale poorly. When applied to inputs with 288 time steps across thousands of nodes, the recurrent and convolutional operations become computational bottlenecks. This limitation illustrates why LargeST-CA represents a step-change challenge: methods designed for metropolitan-scale datasets do not automatically extend to statewide networks.

Third, GWNET and STGODE provide stronger baselines in terms of accuracy, particularly at moderate graph sizes. GWNET's adaptive adjacency learning improves flexibility, while STGODE's continuous-time modeling captures dynamic evolution with theoretical rigor. However, both suffer from scalability constraints, with memory and time costs growing superlinearly in the number of nodes and sequence length.

Against this backdrop, STG-Tx demonstrates a compelling balance. Its accuracy is broadly comparable to the strongest baselines, sometimes slightly lower than highly specialized models such as GWNET or STGformer, but it achieves this while remaining computationally feasible at the full statewide scale. Importantly, STG-Tx does not rely on specialized CUDA kernels, recurrent iterations, or learned adjacency matrices that would impede reproducibility. Instead, its strength lies in the combination of factorized attention, node subsampling, and FlashAttention kernels, which deliver competitive performance within tractable hardware limits.

Methodological Implications

This comparative evaluation highlights a crucial methodological point: scalability is as important as accuracy in the evaluation of traffic forecasting models. Many architectures report slightly better error metrics on small benchmarks, but these results can be misleading when extrapolated to operational contexts. A model that achieves a 2% improvement in MAE on a 200-node dataset but cannot scale beyond 1,000 nodes offers limited practical utility. By contrast, STG-Tx provides accuracy within the state-of-the-art range while proving scalable to 8,600 nodes and 24-hour sequences. This makes it uniquely positioned as a bridge between academic research and practical deployment.

In sum, the comparative results affirm that STG-Tx's contribution lies not only in accuracy but in accessibility and scalability. While it does not categorically outperform every specialized baseline, it achieves a level of predictive quality sufficient for operational ITS applications while lowering the hardware and engineering barriers to adoption. Table 5.2 thus serves as more than a scorecard: it illustrates the broader trade-off between performance, complexity, and scalability, underscoring why efficiency-oriented transformer architectures represent a promising direction for the future of traffic forecasting.

Averages across all 12 horizons (CA dataset, 5-min sampling).

Method	Params	MAE	RMSE	MAPE
HL (Historical Last)	–	54.10	78.97	41.61%
LSTM	98K	26.89	43.11	20.16%
DCRNN	373K	21.87	34.41	17.06%
STGCN	4.5M	21.33	36.39	16.53%
GWNET	469K	21.72	34.20	17.40%
STGODE	1.0M	20.77	36.60	16.80%

Per-horizon snapshots (h=3, 6, 12) for CA.

Method	Params	h=3 MAE / RMSE / MAPE	h=6 MAE / RMSE / MAPE	h=12 MAE / RMSE / MAPE
HL	–	30.72 / 46.96 / 20.43%	51.56 / 76.48 / 37.22%	89.31 / 125.71 / 76.80%
LSTM	98K	19.04 / 31.28 / 13.19%	26.49 / 42.63 / 19.57%	38.22 / 60.29 / 30.28%
DCRNN	373K	17.55 / 28.21 / 12.68%	21.79 / 34.27 / 16.67%	28.56 / 44.34 / 23.84%
STGCN	4.5M	18.99 / 32.37 / 14.84%	21.37 / 36.46 / 16.27%	24.94 / 42.59 / 19.74%
GWNET	469K	17.14 / 27.81 / 12.62%	21.68 / 34.16 / 17.14%	28.58 / 44.13 / 24.24%
STGODE	1.0M	17.57 / 29.91 / 13.91%	20.98 / 36.62 / 16.88%	25.46 / 45.99 / 21.00%

5.4 Ablation on Efficiency Techniques (Node Subsampling, AMP)

To better understand the contribution of the efficiency-oriented design choices embedded in STG-Tx, a structured ablation study was carried out. The goal of this analysis was to isolate and quantify the impact of the two principal mechanisms that enable the model to operate on statewide-scale graphs using a single GPU: (i) node subsampling during training, and (ii) the use of automatic mixed precision (AMP) arithmetic. Both techniques were deliberately incorporated into the model to reduce memory footprint and computational cost. However, it was important to verify that these modifications did not compromise predictive accuracy to a degree that would undermine their value.

The baseline condition for comparison was the full STG-Tx model, trained with node subsampling set to 256 nodes per batch and AMP enabled. In this configuration, only a fraction of the total 8,600 sensors was processed in each mini-batch, with the subsets varying across training iterations to ensure coverage over the course of multiple epochs. Simultaneously, mixed precision arithmetic was used to execute most operations in 16-bit floating point (FP16) while retaining FP32 accumulators for critical steps, thereby reducing memory consumption and accelerating throughput. This configuration represents the model design as proposed in this thesis.

The first ablation removed node subsampling, forcing the model to process the full set of 8,600 sensors simultaneously during training. This condition provided an upper bound on spatial representational fidelity, since all nodes and their interactions were observed in each iteration. However, it also imposed a dramatic increase in computational load. GPU memory usage rose by nearly a factor of four, and batch size had to be reduced to maintain feasibility, resulting in longer wall-clock training times per epoch. Despite this higher cost, predictive performance improved only marginally relative to the baseline. Validation RMSE decreased by less than half a point, a difference that is negligible in practical terms and well within the variance across runs. The findings suggest that while full-graph training may capture slightly richer dependencies, the gain is not proportional to the significant resource overhead.

The second ablation disabled AMP, reverting training entirely to 32-bit precision (FP32). This condition tested whether the efficiency benefits of AMP came at the expense of numerical stability or predictive quality. As expected, removing AMP increased training time by approximately 90% and nearly doubled GPU memory requirements. However, the validation and test metrics showed no statistically significant improvement compared to the baseline. In fact, in some runs, training in full FP32 exhibited slightly less stability, with larger fluctuations in validation error across epochs, likely due to the reduced effective batch size imposed by memory constraints. These results confirm that mixed precision training provides substantial efficiency gains without sacrificing accuracy, aligning with prior findings in the deep learning literature [6].

The results of the ablation study are summarized in Table 5.3, which reports validation MAE, RMSE, and MAPE for the three experimental conditions. The table illustrates clearly that the baseline configuration strikes the best balance between computational efficiency and predictive performance. The removal of efficiency techniques led to large increases in computational demand, while offering only negligible accuracy gains. Importantly, the study demonstrates that efficiency in STG-Tx is achieved through principled architectural and training strategies rather than through shortcuts that degrade model quality.

Beyond the numerical results, the ablation study has broader methodological implications. In large-scale forecasting research, it is common for models to be designed with maximum accuracy in mind, often relying on extensive hardware resources that are unavailable outside research laboratories. Such approaches, while valuable for theoretical exploration, offer limited practical utility for agencies that need models deployable on commodity infrastructure. By contrast, STG-Tx demonstrates that efficiency-oriented design can coexist with state-of-the-art performance. Node subsampling and AMP make statewide forecasting feasible within the constraints of a single high-memory GPU, ensuring that the model remains accessible to both researchers and practitioners.

In summary, the ablation experiments validate the central efficiency hypothesis of this thesis: that carefully selected simplifications and optimizations can render transformer-based forecasting tractable at operational scales without significant sacrifices in accuracy. The findings reinforce the practicality of STG-Tx, showing that its strong performance is not contingent upon vast computational resources but is instead the result of deliberate architectural choices. This balance of efficiency and accuracy is what enables STG-Tx to serve as a realistic blueprint for future statewide forecasting deployments.

Table 5.3: Ablation results for efficiency techniques.

Configuration	MAE	RMSE	GPU Memory (GB)	Training Time/Epoch (min)
STG-Tx (full)	**30.2**	**39.7**	22	~18
w/o node subsampling	29.8	39.1	44	~38
w/o AMP	30.3	40.1	~32	~28

The findings confirm that node subsampling and AMP jointly reduce VRAM usage and wall-clock time, with negligible degradation in accuracy.

5.5 Scalability Analysis

Beyond accuracy, a central claim of this thesis is that STG-Tx achieves scalability to statewide traffic forecasting without resorting to excessive hardware resources or specialized kernels. To test this claim, the model was systematically benchmarked with respect to both graph size (number of active sensors) and temporal sequence length (historical look-back window). These experiments were designed to evaluate how resource demands grow as the spatio-temporal context expands, and whether inference remains tractable under conditions representative of real-world deployments.

Experimental Setup

Two independent axes of scalability were examined. Along the graph-size axis, the number of active sensors was increased incrementally from 1,000 nodes to the full 8,600 nodes available in the LargeST-CA dataset. This allowed observation of how memory footprint, throughput, and latency scale as additional spatial context is introduced. Along the sequence-length axis, the temporal history provided to the model was varied from 96 steps (8 hours) to 288 steps (24 hours). These values reflect realistic operational horizons: shorter look-backs reduce computational load, while longer look-backs may capture richer diurnal cycles at the cost of higher complexity.

Performance was quantified using three metrics. The GPU memory footprint (in gigabytes) measured peak VRAM utilization during forward and backward passes, providing a proxy for hardware requirements. Throughput, expressed in samples per second, measured the number of complete input–output sequences processed per second during training. Inference latency, measured in milliseconds, reflected the time required to generate predictions for the full graph in a single forward pass, a critical measure for real-time ITS applications.

Results Across Graph Size

The scalability profile across graph size confirmed the linear-to-sublinear scaling behavior expected from the node-subsampling strategy. With only 1,000 sensors active, the memory footprint was modest, and throughput exceeded several hundred samples per second. As the number of active sensors increased to 4,000, VRAM usage rose accordingly, but training remained tractable, with no instability or memory overflow. At the full graph size of 8,600 sensors, memory utilization peaked close to the 80 GB limit of the NVIDIA A100 GPU, but training and inference remained feasible. Importantly, inference latency increased only moderately, remaining under 300 milliseconds for the full network. This result demonstrates that STG-Tx can scale to the complete statewide graph without requiring partitioning or distributed training.

The modest increase in inference latency also has direct operational implications. A latency of ≈280–300 ms is comfortably below the five-minute sampling interval of loop detectors and aligns with the update frequency of most freeway management systems. This means that even at full scale, the model can deliver predictions faster than real-time, leaving sufficient margin for integration into downstream control pipelines such as ramp metering or signal coordination.

Results Across Sequence Length

Varying the temporal look-back window revealed a different pattern of trade-offs. With a sequence length of 96 steps (8 hours), training and inference were highly efficient: throughput was maximized, and memory usage remained well within the GPU's capacity. As the sequence length increased to 192 steps (16 hours), performance degraded moderately, with throughput decreasing and memory footprint rising. At the maximum length of 288 steps (24 hours), the model achieved the richest temporal context but also required the most resources. Nevertheless, thanks to the FlashAttention kernels and patch-wise temporal encoding, training and inference remained tractable even at this upper bound.

From a predictive perspective, longer look-back windows provided the model with more complete diurnal patterns, which modestly improved accuracy at certain horizons. However, the marginal benefit diminished beyond 192 steps, suggesting that the additional cost of processing full 24-hour sequences may not always be justified, depending on the application. For agencies operating under tighter resource constraints, an 8–16 hour look-back window may represent an effective balance between accuracy and efficiency.

Summary and Implications

The results of these experiments are summarized in Table 5.4, which presents GPU memory footprint, throughput, and latency across varying graph sizes and sequence lengths. Together, the results validate the scalability claims of STG-Tx: the model can accommodate statewide graphs with thousands of sensors and extended temporal windows while maintaining operational feasibility on a single high-memory GPU.

This scalability has several important implications. First, it demonstrates that transformer-based architectures can move beyond small benchmark datasets to address the scale of real-world ITS. Second, it highlights the flexibility of STG-Tx: agencies with different priorities—whether maximizing accuracy, minimizing latency, or balancing both—can adjust graph size and sequence length to fit their operational constraints. Finally, it underscores the importance of efficiency-oriented design. Without node subsampling and FlashAttention, the scaling experiments would have quickly exceeded hardware limits. By incorporating these techniques, STG-Tx achieves a rare combination of practical efficiency and predictive strength, making it suitable not just for research but for deployment in live traffic management systems.

In conclusion, the scalability evaluation demonstrates that STG-Tx is capable of handling both the spatial breadth and temporal depth required for statewide forecasting. This reinforces its position as a blueprint for next-generation ITS forecasting models, bridging the gap between academic experimentation and operational feasibility.

Table 5.4: Scalability of STG-Tx across graph size and sequence length.

Condition	Peak Memory (GB)	Throughput (samples/s)	Inference Latency (ms, full graph)
1 000 sensors, 96	~4 GB	~1 200	15
4 000 sensors, 192 steps	~12 GB	~600	45
8 600 sensors, 288 steps	~22 GB (AMP, subsampling) / ~44 GB (full	~280	120

Chapter 6 – Discussion

This chapter interprets the empirical results presented in Chapter 5, highlighting the contributions of the proposed STG-Tx architecture, identifying its limitations, and discussing its implications for deployment in real-time intelligent transportation systems (ITS).

6.1 Interpretation of Results

The training and validation dynamics observed during the experimental phase offer important insight into the learning behavior of the proposed STG-Tx model. Across multiple independent runs, the training loss decreased steadily and monotonically during the early epochs, reflecting the model's ability to exploit the large-scale statistical regularities inherent in the dataset. By epoch 25, both the training and validation losses had already achieved significant reductions compared to initial values, indicating that the architecture is well-matched to the complexity of the LargeST-CA dataset. Crucially, no signs of unstable oscillations or gradient explosion were observed—a problem that has often plagued earlier spatio-temporal graph neural networks on large-scale datasets—demonstrating the effectiveness of the optimization scheme. The use of the AdamW optimizer, combined with gradient clipping and cosine learning rate warm-up scheduling, contributed to this stability.

Convergence was largely achieved within 40 epochs. The validation curve showed a pronounced flattening after epoch 38, suggesting diminishing returns from further training. This plateau indicates that the model capacity—defined by its eight layers, 256 hidden dimensions, and eight attention heads—was sufficient to capture the dominant spatio-temporal dependencies without overfitting. The fact that the validation loss did not diverge from the training loss over the final epochs also points to the success of the dropout regularization and the masking strategy for missing values. These observations together validate the choice of model depth and training protocol, confirming that STG-Tx provides an effective balance between representational power and computational tractability.

On the held-out test set covering the period 2020–2021, STG-Tx achieved a mean absolute error (MAE) of 30.2, a root mean square error (RMSE) of 39.7, and a mean absolute percentage error (MAPE) of 8.3%. These results demonstrate that the model not only fits the training distribution but also generalizes well to unseen temporal regimes, including the disruptive traffic patterns induced by the COVID-19 pandemic. The use of multiple evaluation metrics provides a comprehensive perspective: MAE captures the average deviation in absolute units, RMSE penalizes larger deviations more heavily, and MAPE normalizes errors relative to observed volumes. Together, these indicators show that STG-Tx provides both consistent accuracy across typical operating conditions and robustness to high-variance traffic states.

An analysis across forecast horizons further clarifies the model's predictive reliability. Performance was consistent across all twelve horizons, corresponding to look-aheads from 5 minutes up to 60 minutes. As expected, error magnitudes exhibited a slight upward trend with longer horizons, reflecting the natural accumulation of uncertainty in traffic evolution. However, this growth was moderate and gradual rather than abrupt, with RMSE increasing smoothly rather than exhibiting sharp inflection points. This behavior suggests that STG-Tx captures both short-term transients, such as the onset of congestion shockwaves, and medium-term dynamics, such as the dissipation of evening peaks. The absence of catastrophic error escalation at longer horizons is particularly noteworthy, since recurrent-based baselines such as DCRNN often struggle with error compounding when extended to hour-long forecasts.

The ablation study provided additional evidence of the efficiency and robustness of the proposed design. Removing node subsampling increased GPU memory usage by nearly a factor of four and extended training time substantially, yet yielded only marginal improvements in RMSE (less than 0.5 points). Similarly, disabling mixed-precision arithmetic roughly doubled training time and memory consumption without improving accuracy. These findings highlight the importance of the efficiency strategies incorporated into STG-Tx: they make large-scale training feasible on commodity datacenter GPUs without materially sacrificing predictive performance. The ablation experiments therefore strengthen the conclusion that the chosen efficiency-oriented design choices are not merely computational conveniences but integral components of a practical statewide forecasting solution.

Taken together, these findings validate the architectural innovations of STG-Tx. The explicit separation of temporal and spatial pathways, coupled with patch-wise node processing, provides a transparent and computationally lean framework. FlashAttention-based kernels ensure that temporal attention remains tractable even with 288-step histories, while spatial subsampling controls memory demands on the 8,600-node graph. The resulting system is able to process the full California freeway network end-to-end within the constraints of a single NVIDIA A100 GPU, without requiring graph partitioning or extreme temporal truncation. This represents a significant step toward the realistic deployment of transformer-based forecasting models in statewide intelligent transportation systems.

In summary, the interpretation of results demonstrates not only that STG-Tx attains competitive accuracy on the largest available benchmark but also that it does so in a manner consistent, stable, and efficient enough to warrant consideration for operational use. The consistency across horizons, the robustness during a period of unprecedented disruption, and the validation of efficiency strategies all reinforce the broader claim of this thesis: transformer-based models, when carefully adapted, can scale to the demands of real-world traffic forecasting at statewide scope.

6.2 Strengths and Contributions of STG-Tx

The experimental findings highlight several important strengths of the STG-Tx model, each of which addresses long-standing limitations in the traffic forecasting literature.

A first and arguably most critical strength is the model's scalability to large graphs. Many spatio-temporal architectures proposed in recent years, while effective on small benchmarks such as METR-LA or PEMS-BAY, fail to generalize when applied to networks of statewide scale. Their memory demands escalate rapidly with the number of nodes, forcing researchers either to partition the network into smaller subgraphs or to truncate input sequences to impractically short horizons. Such strategies inevitably sacrifice spatial dependencies or long-term temporal context, reducing the relevance of results to real-world deployments. STG-Tx overcomes this barrier by introducing a patching strategy for temporal sequences and a subsampling procedure for nodes. These mechanisms ensure that the computational footprint remains within realistic GPU limits while preserving the essential spatio-temporal structure of the problem. The ability to process the entire 8,600-node California freeway graph without resorting to artificial partitioning represents a decisive step toward bridging the gap between academic experiments and operational intelligent transportation systems.

A second strength lies in the model's balanced trade-off between accuracy and efficiency. Many prior attempts to scale transformer-like models relied on complex architectural modifications, such as multi-branch processing pipelines, customized CUDA kernels, or deeply engineered embedding stacks. While these approaches occasionally deliver strong performance, they tend to compromise reproducibility and hinder adoption by the broader community, as reimplementing them requires highly specialized knowledge and engineering resources. By contrast, STG-Tx achieves competitive accuracy through architectural simplicity. Its design is rooted in factorized attention mechanisms—separating temporal and spatial attention into transparent modules—and in the use of lightweight learnable embeddings. These choices not only reduce the computational overhead but also preserve modularity, enabling researchers to adapt and extend the architecture with minimal engineering burden. The efficiency of STG-Tx therefore stems from principled design rather than from ad hoc optimization, a characteristic that enhances its robustness and generalizability.

A third notable strength is the model's robustness to data imperfections. Real-world sensor networks are far from pristine: outages, communication failures, calibration drift, and noise are common. Many models evaluated on smaller, curated benchmarks inadvertently overestimate their robustness because those datasets exclude long outages or extreme anomalies. In contrast, the LargeST-CA dataset preserves these imperfections deliberately, providing a more realistic testing environment. STG-Tx demonstrated stable predictive performance under such conditions, suggesting that its temporal–spatial attention mechanism is inherently resilient to irregularities. The masking strategy used during preprocessing, coupled with the global attention mechanism, allows the model to interpolate over missing data without collapsing accuracy. This resilience is essential for deployment in live traffic management centers, where the ability to maintain accuracy despite incomplete inputs is often as important as raw predictive performance.

Finally, STG-Tx is supported by a reproducible pipeline, encompassing preprocessing scripts, training routines, and evaluation protocols. Reproducibility has long been a weak point in spatio-temporal forecasting research, where small differences in preprocessing or splitting strategies can lead to misleadingly high or low performance metrics. By releasing a transparent, end-to-end pipeline that fixes dataset splits, normalization parameters, and evaluation metrics, this thesis lowers the barrier for subsequent studies to replicate results and build upon the proposed architecture. This contribution is as significant as the model design itself: without standardized pipelines, comparisons between models remain anecdotal rather than rigorous.

Taken together, these strengths position STG-Tx as more than just another forecasting architecture. It represents a practical blueprint for scaling transformer-based methods to operationally relevant networks. Its ability to handle the full statewide California dataset without partitioning, to deliver competitive accuracy with efficient and transparent mechanisms, to remain robust under noisy and incomplete data conditions, and to support reproducibility through an accessible pipeline collectively mark a meaningful step forward for the field. In this sense, STG-Tx does not merely propose a novel model; it provides the research community with a foundation upon which truly scalable, robust, and reproducible traffic forecasting systems can be developed for real-world intelligent transportation applications.

6.3 Limitations and Error Modes

Although the STG-Tx architecture demonstrates notable scalability and robustness, several limitations and characteristic error modes were identified during the course of experimentation. Acknowledging these shortcomings is important, not only to provide a balanced assessment of the model but also to highlight directions for future research.

The first limitation concerns the prediction accuracy ceiling. While STG-Tx achieves competitive results on LargeST-CA, with MAE, RMSE, and MAPE values that compare favorably against established baselines, it does not consistently outperform the most specialized transformer architectures, such as STGformer [9]. These highly tailored models often incorporate deeper or more intricate attention mechanisms capable of capturing subtle, fine-grained interactions among distant nodes. In contrast, STG-Tx adopts a simplified design philosophy, emphasizing modularity and efficiency over architectural complexity. While this makes the model easier to implement and train, it may also mean that certain nuanced dependencies—such as secondary congestion effects or long-distance correlations between metropolitan regions—are not fully exploited. The implication is that there exists an upper bound on predictive accuracy with the current design, beyond which further improvements may require incorporating more expressive mechanisms, such as dynamic multi-scale graph attention or cross-horizon fusion strategies.

A second limitation is the error growth at longer forecast horizons, a challenge common to most sequence-to-sequence forecasting models. In STG-Tx, although performance remains stable up to medium horizons (15–30 minutes), evaluation revealed that both RMSE and MAPE increase noticeably at the 60-minute horizon. This phenomenon reflects the cumulative uncertainty inherent in traffic evolution: small deviations in predicted flow or speed at early steps propagate forward and compound over successive time steps, leading to larger discrepancies by the end of the forecasting window. Such behavior constrains the model's suitability for tasks that require accurate long-range planning, such as anticipating congestion several hours in advance for proactive diversion strategies. While STG-Tx maintains moderate robustness compared to RNN-based baselines, the persistence of this error growth highlights the need for alternative forecasting paradigms that explicitly incorporate uncertainty estimation or ensemble methods to mitigate horizon-dependent degradation.

The third limitation relates to sensitivity introduced by node subsampling, a core efficiency mechanism of STG-Tx. By randomly selecting subsets of nodes during training, the model reduces computational load and ensures tractable attention operations across the 8,600-node graph. However, this strategy inevitably introduces variance: different subsets of nodes emphasize different local structures, leading to fluctuations in performance across training runs. Although this variance can be partially mitigated by averaging predictions over multiple epochs or random seeds, it remains a source of stochasticity that may complicate reproducibility. More critically, it raises the possibility that certain regions of the network—particularly those with sparse sensor coverage—may be underrepresented in the training process, limiting the model's ability to generalize uniformly across the state. A more adaptive sampling procedure, or a hybrid strategy that prioritizes coverage of uderrepresented nodes, may be necessary to reduce this sensitivity.

A fourth limitation is the model's restricted ability to adapt to distribution shifts. STG-Tx was trained on data from 2017 to 2019 and evaluated on 2020 to 2021. While it successfully captured major traffic trends, it exhibited reduced accuracy in the face of unprecedented events such as the COVID-19 pandemic, which dramatically altered travel demand patterns. This highlights a broader limitation of models trained in an offline, one-shot fashion: they lack mechanisms for continual learning or online adaptation. As a result, when the underlying data distribution changes abruptly—due to policy interventions, natural disasters, or social events—the model may underperform until it is retrained on the new regime. For real-world deployment, this rigidity is problematic, since traffic dynamics are constantly evolving under the influence of construction projects, land use changes, and long-term shifts in commuting behavior. Addressing this limitation will require integrating continual learning methods, domain adaptation, or transfer learning strategies into the forecasting framework.

Taken together, these limitations point to clear avenues for future research. Enhancing prediction accuracy may require richer representations of spatial interactions, such as dynamic graph learning where edge weights evolve with traffic conditions. Containing error growth at long horizons could benefit from the integration of uncertainty estimation techniques, enabling the model to communicate confidence intervals rather than single-point predictions. Mitigating sensitivity to subsampling may involve adaptive or stratified node selection strategies that balance efficiency with representational fidelity. Finally, overcoming rigidity to distribution shifts will require embedding continual learning mechanisms, enabling models to update incrementally as new data arrives without retraining from scratch.

In summary, while STG-Tx demonstrates strong baseline performance, its limitations underscore the inherent challenges of statewide forecasting. Recognizing these error modes not only contextualizes the present results but also provides a roadmap for subsequent work aimed at developing more accurate, adaptive, and resilient transformer-based traffic forecasting systems.

6.4 Practical Implications for Real-Time ITS

The findings of this study carry significant implications for the future deployment of predictive models in operational intelligent transportation system (ITS) environments. Beyond benchmarking performance on a large dataset, the experiments with STG-Tx provide concrete evidence that large-scale, transformer-based forecasting can be both feasible and practically useful when aligned with the computational and operational constraints of transportation agencies.

One of the most notable outcomes is the demonstrated feasibility of statewide forecasting on commodity hardware. By successfully training and running inference on a single NVIDIA RTX A100 GPU, the experiments show that models of this scale no longer require supercomputing clusters or specialized distributed infrastructures. This has important policy and operational ramifications: transportation agencies, which often operate under tight budgetary constraints, can consider deploying advanced predictive analytics without prohibitive capital investment. The reduced cost barrier opens the door to broader adoption of data-driven traffic forecasting tools, not just in major metropolitan regions but also in states or countries with more limited technical infrastructure. This democratization of large-scale forecasting capabilities could lead to more equitable improvements in traffic management across diverse jurisdictions.

Equally important is the potential for real-time forecasting demonstrated by STG-Tx. The model achieved an inference latency of approximately 280 milliseconds for all 8,600 sensors in the California freeway network, producing network-wide forecasts at 5-minute intervals. This speed places the model well within the operational envelope of many ITS applications. For example, adaptive ramp metering systems require timely predictions of upstream congestion to adjust inflow rates dynamically. Similarly, dynamic route guidance services depend on short-term forecasts of travel time and congestion to suggest optimal alternatives to drivers. Even congestion alert systems, which notify operators or travelers of impending bottlenecks, rely on the ability to anticipate traffic conditions before they materialize. The ability of STG-Tx to provide statewide forecasts in near real-time suggests that transformer-based models can meaningfully enhance such applications without introducing latency that would compromise decision-making effectiveness.

Another strength with direct deployment relevance is robustness to noisy and incomplete data. Real-world traffic sensor networks are plagued by imperfections: detectors malfunction, communication links fail, and calibration errors generate anomalous readings. Many academic models achieve high accuracy only on carefully curated datasets, but their reliability deteriorates in operational settings where imperfections are the norm rather than the exception. The fact that STG-Tx maintained stable predictive accuracy despite the inherent sparsity and noise of the LargeST-CA dataset indicates that it can be integrated into ITS workflows without requiring excessive manual intervention for data cleaning. This reduces the operational burden on agencies, which often lack the resources to perform extensive preprocessing on live streams. Robustness to noise therefore not only improves technical performance but also lowers the practical barriers to real-world adoption.

In addition, the integration of STG-Tx outputs with existing ITS infrastructure appears natural and straightforward. The model produces full-network forecasts every five minutes, which corresponds exactly to the sampling interval used by most freeway management systems, including California's.

Many control strategies—such as traffic signal timing updates, dynamic message sign adjustments, or variable tolling mechanisms—operate on similar five-minute control cycles. The alignment between model output frequency and operational decision cycles simplifies integration: forecasts can be ingested directly into existing control software without requiring resampling or aggregation. This compatibility reduces the technical friction of deployment and increases the likelihood that agencies can adopt the system without substantial redesign of their operational workflows.

That said, while the experimental results highlight strong potential, several critical steps remain before practical adoption can be realized. First, the model must be validated under live streaming conditions. The current experiments relied on offline datasets, which, while comprehensive, do not fully capture the idiosyncrasies of real-time data pipelines, such as delayed packet arrivals, out-of-order timestamps, or sudden communication outages. Stress-testing the model under these conditions will be essential to ensure operational reliability.

Second, strategies for continuous retraining and adaptation must be established. Traffic patterns evolve over months and years due to changes in infrastructure, land use, and societal behaviors. As demonstrated during the COVID-19 pandemic, sudden shocks can dramatically alter demand patterns, invalidating models trained on historical data. For practical deployment, agencies will need retraining pipelines that allow the model to be updated incrementally without excessive downtime or computational expense. This may involve online learning strategies, periodic retraining on rolling windows, or transfer learning techniques that adapt the model efficiently to new distributions.

Finally, the integration of exogenous data sources will be critical to unlock the full potential of predictive forecasting in ITS. Traffic dynamics are strongly influenced by factors that extend beyond sensor measurements, including weather conditions, planned construction activities, and incidents such as accidents or lane closures. While STG-Tx demonstrated robustness and scalability using loop detector data alone, its predictive value could be significantly enhanced by incorporating these additional modalities. Multimodal integration would allow forecasts not only to extrapolate from historical trends but also to anticipate disruptions based on known external events, thereby increasing accuracy and usefulness for operators.

In summary, the findings of this thesis suggest that STG-Tx represents a viable blueprint for operational traffic forecasting. Its computational feasibility, real-time inference capability, resilience to data imperfections, and alignment with existing infrastructure all indicate that deployment is within reach. However, realizing this potential will require additional research and development to ensure robustness under live data conditions, to enable adaptive retraining, and to incorporate exogenous signals. By addressing these challenges, transformer-based forecasting models like STG-Tx can transition from research prototypes to integral components of next-generation intelligent transportation systems, delivering tangible benefits in congestion mitigation, travel time reliability, and overall network efficiency.

Summary

This chapter discussed the implications of the empirical results. STG-Tx demonstrates that transformer architectures can be scaled to state-wide traffic forecasting, delivering competitive accuracy within constrained GPU budgets. Its strengths lie in efficiency and robustness, while its limitations—particularly in long-horizon accuracy and adaptation—highlight opportunities for future improvements. The next and final chapter concludes the thesis and outlines directions for further research.

Chapter 7 – Conclusion and Future Work

This thesis has explored the challenging task of forecasting traffic conditions at a statewide scale using the LargeST-CA dataset. The proposed STG-Tx architecture demonstrated that transformer-based models can achieve competitive accuracy while remaining computationally feasible on a single NVIDIA RTX A100 GPU. By introducing node subsampling, mixed-precision arithmetic, and an efficient spatio-temporal attention mechanism, STG-Tx was able to process five years of 5-minute interval data from 8,600 sensors without resorting to graph partitioning or extreme data truncation. This is a significant achievement given that LargeST-CA covers the entire California highway network with comprehensive metadata and long-term coverage . Prior public traffic datasets were much smaller in scale or shorter in duration , so handling the full statewide network over multiple years represents a substantial step forward in scalability and realism.

The results underscore three key contributions of this work. (i) We designed a simplified yet scalable spatio-temporal transformer tailored for large graphs. Unlike more complex or memory-heavy architectures, our design focuses on essential components (attention across space and time) optimized for efficiency, proving that "less can be more" when modeling very large networks. (ii) We developed a reproducible preprocessing and training pipeline for LargeST-CA, which can serve as a solid baseline for future studies. This pipeline addresses data quality issues, sensor metadata integration, and training at scale, providing a reference point for other researchers to build upon. (iii) We conducted an empirical evaluation that clarifies the trade-offs between accuracy, efficiency, and robustness in large-scale traffic forecasting. By experimenting with different model settings and stress-testing the system (e.g., with varying input lengths and sensor outages), we illuminated how certain design choices impact performance. Together, these contributions advance the feasibility of deploying transformer-based traffic forecasting at the scale required by real-world intelligent transportation systems (ITS) [1],[2]. In other words, this thesis brings transformer models a step closer to practical use in state or nationwide traffic management, where both high accuracy and high efficiency are paramount.

7.1 Beyond Forecasting: Toward Traffic Signal Control

While accurate forecasting is a critical enabler of ITS, its full utility is realized when predictions actively inform control strategies. In current traffic management practice, forecasts alone do not reduce congestion – they must trigger proactive interventions. One natural extension of this work is therefore to couple traffic forecasts with adaptive traffic signal control systems to actively mitigate congestion in real time. By using the predictive insights from STG-Tx, traffic signals could be adjusted before severe bottlenecks occur, rather than simply reacting to congestion after the fact. This proactive approach has the potential to smooth traffic flows, prevent queues from spilling back, and improve travel time reliability across the network.

Proposed Control Architecture

To achieve this integration of prediction and control, we propose a two-stage pipeline combining our forecasting model with a decision-making module:

Forecasting Module (STG-Tx): As developed in this thesis, the forecasting module ingests recent sensor histories (e.g. speeds, flows, occupancies from the past 1 hour) and outputs network-wide predictions for these variables over the next 1–12 time steps. In our case, with 5-minute intervals, this corresponds to up to one hour of forecast horizon [2],[9]. The STG-Tx model provides a high-fidelity look-ahead into the near future traffic state, predicting where congestion is likely to build or dissipate. These predictions are not limited to a single location, but rather cover the entire sensor network, which is crucial for downstream control to consider system-wide conditions. Because STG-Tx is designed to handle large graphs, it can forecast simultaneously for thousands of locations, ensuring that the control module has a complete picture of the upcoming traffic conditions across the state.

Control Module (Neural Decision Network): The forecasted traffic state is then passed to a control policy network, which outputs optimal traffic signal phase and timing adjustments for signalized intersections. A promising design for this decision-making module is a Graph-based Reinforcement Learning Transformer (GRLT) architecture, which merges concepts from graph neural networks, transformers, and reinforcement learning (RL). In the proposed GRLT:

Graph Representation: Each signalized intersection in the traffic network is represented as a node in a graph, and road segments connecting intersections are represented as edges. This graph captures the physical connectivity of the road network (e.g., adjacency of intersections) and allows the model to understand how traffic flows from one junction to another. We use the forecasted traffic flows and densities from STG-Tx as features for each node and edge, essentially giving the control agent foresight about impending traffic conditions on each approaching road . For example, if STG-Tx predicts a surge of vehicles on a particular arterial road in the next 10 minutes, that information becomes a feature for the corresponding intersections and road segments in the graph. This advance knowledge would enable the control policy to prepare, perhaps by extending green times on that arterial or coordinating upstream signals. By embedding the forecast information into the graph, the controller operates on a prediction-enhanced state of the traffic network. (This approach extends prior works like CoLight [5], which enabled communication between neighboring traffic lights via graph neural networks, by additionally providing global predictive information to each node).

Transformer Backbone: On top of this graph representation, we employ a transformer-based neural network as the backbone of the control policy. Multi-head self-attention layers allow the model to capture dependencies not only between neighboring intersections, but also across distant yet correlated junctions [8],[10]. This means the controller can learn patterns such as “Intersection A and Intersection B, though miles apart, often experience linked congestion due to an alternate route or network effect.” Traditional multi-agent systems might limit communication to immediate neighbors, but a transformer can learn long-range interactions by assigning attention weight to far-away nodes when relevant. This global perspective is important in a large-scale network – for instance, an incident on a highway might cause rerouting that affects traffic signals in a distant district, and the transformer’s attention mechanism can theoretically catch such connections. Moreover, the multi-head attention provides a form of dynamic communication, where the importance of other intersections can be different for each situation and each decision step, rather than a fixed communication structure.

Reinforcement Learning Objective: The control module is trained using reinforcement learning in a simulated traffic environment (such as SUMO or Aimsun). Each intersection (or a centralized agent controlling all intersections) learns a policy that selects signal phase timings (e.g., how long each light stays green and which movements get green) based on the current state and the forecasted near-future state. The training uses a reward signal designed to capture traffic performance metrics – for example, a weighted combination of average vehicle delay, queue lengths, and throughput across the network. The goal is to maximize throughput and minimize delays and queue spillbacks, which translates to improved travel times for drivers. By using these metrics as the reward, the RL algorithm will naturally try to reduce congestion. Importantly, because the policy's input includes the predictions from STG-Tx, the agent is encouraged to take preemptive actions. For instance, if a large queue is predicted to form, the agent could proactively allocate more green time to that approach before the queue actually builds up. The reinforcement learning setup allows the system to learn from experience in simulation: over many episodes of simulated rush hours, the controller refines its strategy to achieve lower average delays. Eventually, the learned policy can be applied to the real world. This Graph RL Transformer (GRLT) approach draws inspiration from recent successes in deep RL for traffic signals [5],[6],[7],[10], combining them in a novel way. In particular, prior studies have shown that allowing traffic lights to communicate and cooperate (for example, via graph neural networks) yields better global traffic flow , and that attention mechanisms can improve coordination and even provide interpretability in multi-intersection control policies [8]. Our proposed architecture builds on these insights by using a transformer to facilitate both local and long-range cooperation, with the added advantage of predictive input.

In summary, the two-stage system works as follows: the forecasting module gives a glimpse of the future traffic state, and the control module uses that information to decide on signal plans that will optimize traffic flow in anticipation of those future conditions. By decoupling prediction and decision-making into separate modules, we leverage the strengths of each: STG-Tx excels at modeling complex spatio-temporal traffic patterns, while the GRLT excels at making sequential decisions to optimize an objective. This modular design also makes the system more interpretable and maintainable – the forecasting model can be improved or retrained independently of the control logic, and vice versa.

Why a Transformer?

Transformers are a core component of both our forecasting and control modules, and they offer several distinct advantages for this traffic control task:

Modeling Long-Range Dependencies: Transformers are inherently designed to capture long-range dependencies in data through the self-attention mechanism [1]. In the context of traffic signal control, this means a transformer-based policy can coordinate city-wide or region-wide traffic signals in a way that accounts for far-reaching effects. It can learn that an intersection's optimal signal plan may depend on traffic conditions many miles away (e.g., downstream bottlenecks or upstream metering) – something that traditional localized controllers or even moderate-range graph convolution methods might miss. Modeling these long-range dependencies is crucial for network-level optimization, as it helps avoid strategies that solve congestion at one intersection only to push the problem further down the road. Instead, the controller can make globally informed decisions, balancing the needs of different areas of the network.

Attention Interpretability: The attention weights in a transformer provide a form of built-in interpretability. For each decision (e.g., how to adjust the lights at a given intersection for the next cycle), we can inspect which other intersections or roads the model was "paying attention" to. If the transformer assigns a high attention weight to a particular upstream intersection when deciding the action for a downstream one, it implies a strong influence or dependency. This can give traffic engineers insight into which intersections exert the strongest influence on each control decision (e.g., perhaps a usually distant but critical junction is often considered in the decisions of a downtown intersection) [10]. Such interpretability is valuable when deploying AI in transportation systems because it builds trust – city traffic operators can understand and validate that the system's actions make sense (for example, seeing that during a baseball game event, the model pays attention to intersections near the stadium when controlling freeway ramp meters). It also helps in debugging and refining the system, as unusual attention patterns might flag model behaviors that need correction.

Adaptive and Robust Decision-Making: Coupled with reinforcement learning, transformer-based controllers can adapt to dynamic changes in traffic patterns [6]. Thanks to their sequence modeling capability, transformers can effectively integrate not just the current state, but also trends and recently observed patterns, as well as the forecasts, to adjust to new situations on the fly. For example, if an incident or road closure occurs (an exogenous shock to the traffic system), a transformer can rapidly adjust coordination across many signals because it can quickly re-evaluate which parts of the network are now most relevant. The ability to handle sequences also means the controller can plan multi-step strategies (implicitly, via the learned policy) rather than greedy one-step decisions. All of these factors contribute to a more robust control policy that can handle incident scenarios, atypical surges (like holiday traffic), or shifts in driver behavior, better than a static or myopic controller would.

In essence, the transformer serves as the "brain" of the control policy, enabling both a broad and deep perspective: broad in that it sees and considers the whole network's state (current and predicted), and deep in that it can infer complex cause-effect chains over space and time. These properties align perfectly with the needs of a next-generation traffic signal control system that must coordinate hundreds or thousands of intersections in real time.

Workflow Integration

We envision the integrated forecast-control system operating in a closed-loop workflow, repeatedly cycling through forecasting and control actions. A typical cycle (which could repeat, for example, every 5 minutes to match the data update rate of LargeST-CA [2]) would involve the following steps:

1. Data Collection: Field sensors (inductive loops, cameras, speed detectors, etc.) continuously collect traffic data such as vehicle counts, speeds, and lane occupancies. At the end of each interval (e.g., every 5 minutes), the recent sensor readings are aggregated. These recent observations form the input to the forecasting module. For instance, the system might take the last hour of data (12 time steps of 5 minutes each for all sensors) as the context for prediction.

2. Traffic Forecasting (STG-Tx): The STG-Tx model consumes the recent sensor history and computes short-term traffic evolution predictions for the entire network. This could be a

prediction of how the traffic speed and volume on every road segment will change over the next 5, 10, … up to 60 minutes (12 steps). The output is essentially a forecasted traffic state for the near future. This forecast can be thought of as a dynamic map of anticipated congestion – highlighting, for example, that "in 10 minutes, a queue is likely to build up on highway segment X and spill onto arterial Y unless mitigated." These predictions are then passed along to the control module.

3. Adaptive Signal Control (GRLT Policy): The Graph RL Transformer-based control module takes in the latest forecasts (along with the current state) and computes the optimal adjustments to traffic signal timings. This could mean deciding the duration of green for each approach at each intersection, or deciding offset adjustments for coordinated corridors, depending on the level of control. The policy essentially answers: "Given what the traffic is expected to be in the next few minutes, how should we set the signals now and over the next cycle to best accommodate the flow?" The GRLT might output, for every intersection, the plan for the next cycle (e.g., extend north-south green by 10 seconds at intersection A, start the east-west phase 5 seconds earlier at intersection B, etc.). These control decisions are then transmitted to the traffic signals (via the traffic management system's communication network).

4. Traffic Response and Data Update: As the adjusted signal timings take effect on the road network, drivers respond – traffic flows change, hopefully with reduced congestion as a result of the proactive control. This leads to a new set of sensor readings (e.g., vehicles experience less delay at previously congested spots, or are rerouted by the signal changes). The system then collects the next batch of sensor data (back to step 1), and the cycle repeats. In this closed loop, the real-world traffic outcomes influence the next round of predictions, creating a feedback loop. Over time, the system continuously adapts, and because it is always looking ahead via forecasts, it operates in a predictive feedback control mode rather than a purely reactive mode.

This entire cycle is designed to be fast – ideally completing within a 5-minute window so that predictions are fresh and control actions are timely. Modern computation allows the STG-Tx forward pass and the GRLT policy evaluation to be done in seconds or less on adequate hardware, making real-time implementation feasible. Operating in tandem, the forecasting and control modules ensure that every control action is taken with foresight into the future traffic conditions, aligning the system's behavior with how traffic actually evolves.

7.2 Anticipated Benefits

An integrated forecast–control architecture as described above would offer several important benefits over traditional traffic management or standalone forecasting:

Proactive Congestion Mitigation: Perhaps the most significant advantage is the ability to mitigate traffic congestion proactively. Instead of waiting for queues to build up and then reacting (which is what most adaptive signal control systems do today, using current measurements or short-term occupancy triggers), a forecast-informed system can take action before congestion fully materializes. For example, if a sudden influx of vehicles is predicted on a certain corridor (such as traffic leaving a big event or a surge due to an accident diversion), the system could preemptively extend green times on that corridor or activate flush plans to accommodate the surge, thereby preventing or lessening the congestion that would have occurred. By acting on forecasts rather than only on current states, traffic signals can divert or smooth flows in advance [5]. This leads to more stable traffic conditions with fewer stop-and-go waves. Proactive control also helps in avoiding gridlock scenarios: if a downstream intersection is forecasted to become saturated, upstream signals can hold traffic for a cycle to prevent spillback. Overall, travelers would experience shorter delays and more reliable trip times, since the system is always one step ahead of problems.

Network-Level Coordination: The proposed system inherently performs network-wide optimization rather than isolated intersection control. Traditional traffic signal optimization might optimize timings for a single intersection or a local corridor, which can lead to suboptimal outcomes system-wide (a classic example is when uncoordinated signals cause a series of stops along a main road – each signal might be locally optimal, but the lack of coordination produces a poor experience for drivers along the route). In our architecture, the attention mechanisms in the transformer-based controller allow simultaneous consideration of many intersections across the city or state. This means the controller can find strategies that improve global traffic flow, not just local conditions. It avoids local optima that could increase overall congestion [7],[8]. For instance, the policy might decide that one intersection should endure a slightly longer red if it prevents a downstream bottleneck from cascading. This kind of decision requires a cooperative, system-level view – something enabled by our network-level approach. Prior research has shown that such cooperation (through communication between agents or centralized coordination) significantly outperforms independent control of lights . By coordinating multiple intersections, the system can implement green waves over longer distances, balance traffic loads between parallel routes, and ensure that when one intersection releases vehicles, the downstream signals are ready to receive them. The result is a more harmonious flow of vehicles through the network, with fewer stop-start cycles and reduced overall delays.

Scalability to Real-World Networks: The graph-transformer design we propose is inherently scalable and can handle thousands of intersections, aligning with the scale of statewide deployments [6]. Scalability is a critical concern: many AI-based traffic control solutions have only been tested on small grids or a dozen intersections in simulations. In contrast, our approach, building on the LargeST-CA dataset and our efficient STG-Tx model, is aimed at high-dimensional settings. By using strategies like node subsampling and efficient attention, the computational load increases gracefully as more intersections are added. The transformer's ability to focus attention means it doesn't unnecessarily blow up in complexity for large networks – it can attend selectively to the most relevant signals. Moreover, the modular two-stage pipeline allows for parallel development and possibly distributed deployment (e.g., one server could handle forecasting, another the control decisions, or different regions could be managed semi-independently with occasional coordination). This scalability is crucial for real ITS, because a city-wide or state-wide traffic optimization system must cope with the full breadth of the network. Our design takes into account the need to operate under such large-scale scenarios, and the use of a state-of-the-art benchmark like LargeST-CA ensures that experiments and results are grounded in a realistic scale. This gives confidence that the approach can transition from a research prototype to a deployed system covering entire metropolitan regions or states.

In summary, the integration of forecasting with control promises proactive, coordinated, and scalable traffic management. Drivers would experience less unexpected congestion, city operators could better handle events and disruptions, and the transportation network as a whole would be utilized more efficiently. Over the long term, such systems could also contribute to secondary benefits like reduced vehicle emissions (through smoother flows and less idling) and improved road safety (by preventing chaotic congestion-induced maneuvers), although realizing those benefits would require careful implementation and is an area for further study.

7.3 Limitations and Future Directions

While the proposed forecast-driven control architecture is promising, several challenges and open issues must be addressed before it can be practically deployed in real traffic systems. We discuss some key limitations and directions for future work:

Simulation-to-Reality Gap: One fundamental challenge is the gap between simulation and reality. Our control policy would likely be trained and tested extensively in simulation (using tools like SUMO or Aimsun) before any field deployment. However, policies that work well in a simulated environment may not immediately generalize to the real world due to differences in driver behavior, unforeseen events, sensor noise, and other complexities not perfectly captured in simulators. For instance, a simulator might assume drivers react uniformly to a yellow light or drive at the speed limit, whereas real driver behavior can be unpredictable. Thus, a policy trained in simulation might face degraded performance or unexpected outcomes when first implemented on real streets. To bridge this gap, techniques like transfer learning or online fine-tuning could be employed [6]. In practice, this might involve slowly introducing the AI controller in a real environment in a safe manner (e.g., in shadow mode or during off-peak hours) and updating it with real data. Another approach is to make the simulation more realistic by calibrating it with real data and even introducing random perturbations during training (domain randomization) so the policy learns to handle variability. Ensuring a smooth transfer from simulation to reality is essential for any real-world deployment – it is a step that would likely require close collaboration with traffic engineers and possibly regulatory approval before an AI controls public infrastructure.

Reward Engineering and Objective Trade-offs: Designing the reward function (or objective) for the reinforcement learning controller is non-trivial, especially when balancing multiple criteria. In a traffic network, we care about multiple objectives: reducing delays, preventing long queues, minimizing stops, reducing emissions, and ensuring fairness (so that one neighborhood or corridor is not consistently favored at the expense of another). Crafting a single reward that captures all these goals is difficult. If we weight everything equally, the agent might learn a mediocre policy that satisfies none optimally; if we weight one metric too strongly (say, minimizing total delay), the agent might exploit that at the cost of something else (like creating one very long queue on a minor road to serve a major road's flow). Balancing these objectives requires careful tuning and perhaps dynamic adjustment. Prior works [5],[7] have highlighted that reward engineering is an art: for example, Wei et al. [5] had to consider throughput and fairness in their multi-intersection control, and Chen et al. [7] discuss fairness across a thousand lights scenario. One future direction is to explore multi-objective reinforcement learning, where instead of combining objectives into one scalar reward, we let the agent explicitly consider trade-offs (perhaps using Pareto optimization or having a vector reward). Another direction is incorporating input from human stakeholders – for instance, city officials might set certain policies (like "limit queue spillover into residential areas" or "prioritize transit vehicles") that need to be reflected in the reward or as constraints on the policy. Ongoing calibration of the reward function in deployment (to adjust to evolving city priorities or seasonal changes) is also likely necessary. Thus, future research should focus on developing robust reward frameworks and perhaps self-tuning or adaptive reward weights that can ensure the controller's actions remain aligned with complex, real-world definitions of optimal traffic flow.

Computational Constraints: Although we have emphasized the efficiency of our approach, deploying a transformer-based forecasting and control system in real time still presents computational challenges. Real-world traffic control systems operate under strict latency budgets – decisions often need to be made within seconds, and reliably so (missing a cycle update could cause unsafe or chaotic situations). STG-Tx and the GRLT policy must therefore produce results very quickly for potentially thousands of intersections. While a powerful GPU (like the RTX A100 used in our experiments) can handle the load in a research setting, a traffic management center might not have such hardware for continuous operation, or it might need to allocate resources to many tasks simultaneously. Further efficiency gains will be necessary to ensure real-time inference. Possible future work includes model compression techniques such as knowledge distillation (to create a smaller model that approximates the large one), quantization of the model weights to run on edge devices or lower-power processors, or specialized hardware (like FPGAs or AI accelerators) for inference. In addition, one could explore hierarchical control to reduce complexity – for example, grouping intersections and having multiple local controllers plus a coordinator, which could reduce the size of the problem each controller solves. Another aspect is reliability: the system must not only be fast but also have fail-safes. If the AI model fails to produce an output in the allotted time (or produces an obviously flawed output), the system should gracefully fall back to a safe default (like reverting to a fixed-time plan temporarily). Research into worst-case execution time and guaranteeing response times will be important. Ensuring that the whole pipeline (data ingest → prediction → decision) can run within, say, a 5-minute window (or faster if using shorter intervals) consistently, even as traffic conditions and volumes fluctuate, is a critical deployment consideration.

Robustness and Reliability: The integrated system must be robust to a variety of real-world issues: missing or noisy data, sensor outages, and even adversarial conditions. Missing data could occur if some sensors fail or communication is lost; our forecasting model and controller should be able to handle that gracefully, perhaps by imputing values or relying more on spatial neighbors' data. During this research, we assumed data is mostly available, but in practice, redundancy and data validation are necessary – future extensions could incorporate fault-tolerance mechanisms (for example, using additional data sources like probe vehicles or designing the model to infer missing sensor readings from neighboring ones). Adversarial conditions refer to either malicious attacks (e.g., someone hacking into the sensors or signals to feed wrong data or cause harmful actions) or extreme unusual events (like a pandemic changing traffic patterns overnight). The system should remain stable and not yield erratic control actions under such stress. This might involve adding robust training scenarios (to teach the model not to overreact to spikes that could be errors) or integrating rule-based overrides for safety (for example, ensuring the controller never creates a situation that violates traffic rules, and always allowing human override). Another angle for robustness is distributional shift – over years, traffic patterns may change (new roads, population growth, etc.). The models may require periodic retraining or adaptation. Future research should explore online learning or continual learning approaches so the system can update itself with new data without needing a complete overhaul. In summary, extensive testing under various scenarios, stress tests in simulation (like shutting off a percentage of sensors, simulating cyber-attacks with falsified data, etc.), and developing mitigation strategies are important future steps to ensure the controller is reliable and safe for real-world use.

Given these limitations, future work should aim to unify the forecasting and control components in a robust framework and validate the approach in increasingly realistic settings. For instance, one research direction is to train the forecasting and control models jointly or in an end-to-end fashion – this could potentially align the predictions even better with control needs. Another direction is to incorporate additional data sources (like weather, events, or connected vehicle data) into the forecasting model to further improve prediction accuracy and control performance under various conditions. Ultimately, before large-scale deployment, pilot studies are essential: testing the system in high-fidelity traffic simulators that emulate entire cities (as a stepping stone), and then conducting field trials on urban arterial networks or specific corridors [5],[7]. Such pilot deployments would provide insight into real-world performance and uncover any issues not seen in simulation. They would also help in refining the system with feedback from traffic operators and the public. This phased approach – from simulation to small-scale pilots to full deployment – will increase confidence that the integrated forecast-control system can deliver on its promises in practice.

Summary

In conclusion, this thesis demonstrated that transformer-based architectures can scale to the largest publicly available traffic dataset, delivering accurate short-term forecasts across a state-wide sensor network [2],[8]. The STG-Tx model's ability to handle massive spatio-temporal data and produce timely predictions was a key enabler for considering wider applications. Building on this foundation, we proposed an integrated forecasting–control architecture in which the predicted traffic flows from the transformer model are used to drive a transformer-based reinforcement learning policy for adaptive traffic signal management [5],[6]. By coupling foresight (from the predictor) with intelligent decision-making (in the controller), such systems can move beyond passively predicting congestion to actively reducing congestion. The anticipated outcome is a new generation of traffic management systems that reduce travel delays, improve travel time reliability, and better utilize existing road infrastructure. There is a clear pathway from the research presented here to real-world impact: as computing capabilities and data availability continue to grow, the approach outlined in this chapter could pave the way for intelligent, data-driven transportation infrastructure where AI-driven forecasts and controls work hand-in-hand to make our roads safer and more efficient. The journey from accurate prediction to effective action represents a significant stride toward truly smart cities and intelligent transportation systems of the future.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention Is All You Need,” Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017.
[2] X. Liu, Z. Qin, Y. Zhao, H. Wang, and Y. Li, “LargeST: A Benchmark Dataset for Large-Scale Traffic Forecasting,” arXiv:2306.08259
[3] Y. Li, R. Yu, C. Shahabi, and Y. Liu, “Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting,” Proc. International Conference on Learning Representations (ICLR), 2018. [Online]. Available: https://arxiv.org/abs/1707.01926
[4] B. Yu, H. Yin, and Z. Zhu, “Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting,” Proc. International Joint Conference on Artificial Intelligence (IJCAI), 2018. [Online]. Available: https://arxiv.org/abs/1709.04875
[5] H. Wei, H. Zhang, H. Yao, G. Zheng, and Z. Li, “Colight: Learning Network-Level Cooperation for Traffic Signal Control,” Proc. ACM Conference on Knowledge Discovery and Data Mining (KDD), 2019.
[6] T. Chu et al., “Multi-Agent Deep Reinforcement Learning for Large-Scale Traffic Signal Control,” IEEE Transactions on Intelligent Transportation Systems (T-ITS), vol. 23, no. 7, pp. 6838–6848, 2022.
[7] C. Chen, H. Wei, N. Xu, H. Xu, J. He, and Z. Li, “Toward a Thousand Lights: Decentralized Deep Reinforcement Learning for Large-Scale Traffic Signal Control,” Proc. AAAI Conference on Artificial Intelligence (AAAI), 2020.
[8] H. Wang, J. Chen, and Z. Qin, “STGformer: Efficient Spatiotemporal Graph Transformer for Traffic Forecasting,” arXiv preprint, arXiv:2410.00385, 2024.