Work Plan
(September 2025 - February 2026)
for
Mehmet Cüneyt Özbalcı

Goals and Timeline (September 2025 - February 2026)

Period	Main Goal	Sub-Goals / Deliverables	Evaluation Metrics (Objective & Measurable)
Month 1–2(Literature Review & Dataset Collection)	**Complete comprehensive literature review** and **collect relevant datasets**	- Identify at least 40+ key papers on GNN, dynamic link analysis, traffic modeling, and cross-domain applications (biomedical, citation networks, social media).- Summarize and classify papers (taxonomy by method, dataset, performance, novelty).- Identify gaps to justify the proposed architecture.- Collect and document at least 3+ relevant large-scale traffic datasets with metadata (source, size, format, license).- Preprocess or at least outline preprocessing pipeline.	**Breadth**: Number of peer-reviewed papers reviewed vs. target (e.g., ≥30).**Depth**: Quality of summaries (structured template per paper: problem, method, dataset, results, gaps).**Organization**: Existence of a literature matrix / annotated bibliography.**Gap Analysis**: Clear identification of at least 3–5 gaps.**Dataset Coverage**: Number and size of datasets collected and documented.**Documentation Quality**: Dataset description, licensing, and readiness for use.
Month 3(Conceptual Model Design & Framework Setup)	**Design first-phase model architecture & prepare environment**	- Draft initial GNN architecture diagram (input, feature enrichment module, dynamic link module).- Specify node and edge features; define data schema.- Select baseline models for comparison.- Set up code repository, version control, and environment (PyTorch Geometric, DGL, etc.).	**Architecture Clarity**: Well-labeled architecture diagram with all modules.**Baseline Selection**: Justification and documentation of at least 2 baseline models.**Reproducibility**: Working environment & repository with README, environment.yml/requirements.txt.**Deliverables**: Written document + slides explaining the design.
Month 4(First Phase Implementation)	**Implement node feature enrichment and partial link analysis**	- Implement data loading and preprocessing pipeline.- Implement first module: node feature enrichment layer(s).- Implement initial dynamic link analysis mechanism (may be partial).- Integrate training loop and evaluation metrics (accuracy, AUC, etc., depending on task).	**Code Completeness**: % of planned components implemented (e.g., ≥70%).**Functionality**: Modules run on at least one collected dataset without error.**Version Control**: Commits with meaningful messages.**Preliminary Results**: Baseline performance metrics reported.
Month 5(Testing, Refinement & Preliminary Results)	**Refine and test the first-phase model**	- Evaluate on at least one full dataset.- Compare to baselines (tables/plots).- Identify bottlenecks and propose improvements.- Prepare interim report (background, methodology, preliminary results).	**Evaluation Coverage**: # of datasets tested vs. planned (≥1).**Metrics Reported**: At least 2–3 relevant metrics (F1, AUC, link prediction accuracy, etc.).**Comparison Quality**: Baseline vs. proposed clearly plotted.**Interim Report Quality**: Contains background, methods, preliminary results, tables/figures.

Detailed Plan

Purposes of a PhD-Level Literature Review

Aspect	What It Means for the Student
Comprehensive Coverage	Show you know the full landscape of the field, not just a few papers. Include seminal work + the most recent publications.
Critical Analysis	Not just "what they did" but "how well it works, limitations, gaps, inconsistencies, and where your work fits."
Organization & Synthesis	Group papers by theme, approach, dataset, or method. Show trends and patterns, not a paper-by-paper list.
Gap Identification	Explicitly state open problems and how your work addresses them.
Relevance to Your Research	Tie every section back to your research question or hypothesis.
Scholarly Ethics	Proper citation, no plagiarism, consistent referencing style.

Advisor's Expectation (Deliverables & Standards)

Area	Measurable Standard
Breadth	Reviewed **30–50+** peer-reviewed articles (varies by field) including seminal + last 2–3 years.
Diversity of Sources	Mix of journal papers, conference papers, and key books/technical reports. Avoid relying on only one author or venue.
Currency	Latest research included — at least some papers from the **last 12–24 months**.
Depth of Reading	Each paper summarized in a consistent template (problem, method, dataset, results, limitations).
Synthesis	Created a **taxonomy or table** comparing methods/datasets/performance.
Critical Evaluation	At least 3–5 clear **research gaps** documented.
Writing Quality	Organized sections/subsections, logical flow, and clear transitions.
Referencing	Correct and consistent style (APA/IEEE etc.), no missing references.
Tools	Used a reference manager (Zotero, Mendeley, EndNote) with a shared library for the advisor.
Deliverable	A written draft (or chapter) + summary tables/figures + annotated bibliography.

Best Practices for the Student

1. **Create a shared bibliography** with the advisor (Zotero group or Mendeley group).
2. **Use a structured template** for each paper (problem, method, dataset, results, limitations, relevance).
3. **Build comparison tables** (methods × datasets × metrics).
4. **Include visuals**: timelines of research progress, concept maps of methods.
5. **Do iterative reviews**: update your review monthly as new papers appear.
6. **Highlight gaps explicitly**: e.g., "No existing study uses GNN for dynamic link analysis in traffic networks with node feature enrichment."
7. **Write early**: produce a draft chapter even while still collecting papers.

Evaluation Sheet for *Month 1–2 (Literature Review)*

Metric	0	1	2	3	4	5	Score
Breadth of Coverage (No. & range of sources)	None	<10 papers, narrow scope	10–19 papers, limited scope	20–29 papers incl. some seminal	30–39 papers incl. seminal & recent	≥40 papers incl. seminal, most recent & diverse sources	
Currency (recent research included)	None	All >5 yrs old	Mostly >3 yrs old	Mix old/new but few last 2 yrs	≥25% last 2 yrs	≥50% last 2 yrs & trend analysis	
Depth & Consistency of Summaries (structured template per paper)	None	Very brief/notes only	Some structured summaries	Structured summaries for ≥50%	Structured summaries for ≥75%	Structured summaries for ≥90% + critical notes	
Synthesis / Organization(taxonomy, themes, tables)	None	Disjointed list	Minimal grouping	Basic grouping + 1 table	Clear taxonomy + ≥1 comparison table	Comprehensive taxonomy + visuals (maps/timelines)	
Critical Evaluation & Gap Analysis	None	Descriptive only	Identifies few gaps vaguely	Identifies 2–3 gaps	Identifies 3–5 well-argued gaps	Identifies ≥5 well-argued gaps tied to proposed work	
Relevance to Proposed Research	None	Vague link	Some link to topic	Clear link to topic	Clear link + shows niche	Clear link + positions own contribution explicitly	
Referencing Accuracy(style & completeness)	None	Many errors	Frequent errors	Minor errors	Mostly correct	Fully correct & consistent (APA/IEEE etc.)	
Deliverables & Timeliness (annotated bib, draft chapter, tables)	None	Very late/incomplete	Late/incomplete	On time but incomplete	On time, most deliverables	On time, complete draft + tables/figures	

Core Principles of Data Collection & Preprocessing (PhD Level)

Aspect	What It Means for the Student
Relevance	Data must directly address the research question — no "convenience data" without justification.
Adequacy	Enough size, coverage, and quality to support robust analysis.
Transparency	Clear documentation of data sources, acquisition process, and any transformations applied.
Reproducibility	Another researcher should be able to replicate data acquisition and preprocessing from the description/code.
Ethics & Compliance	Respect copyright, licenses, privacy regulations (e.g., GDPR, IRB approval if human data).
Quality Control	Identification and handling of missing data, noise, duplicates, anomalies.
Suitability for Modeling	Data must be transformed into the right format and feature representation for the planned methods (e.g., GNN).

Advisor's Expectation (Deliverables & Standards)

1. Data Collection

- **Source Identification**: Clear list of data sources (traffic databases, public repositories, sensors, APIs, etc.).
- **Justification of Choice**: Why these datasets? How do they match research goals?
- **Metadata**: Document dataset characteristics (size, variables, time span, sampling frequency).
- **Licensing & Access**: Ensure usage rights and cite properly.
- **Versioning**: If data are updated over time, record which version is used.

Typical Deliverable:

- A **data documentation sheet** (or appendix) with source, description, licensing, size, variables.
- Scripts or code to download/extract data.

---

2. Data Preprocessing

- **Cleaning**: Handle missing values, outliers, inconsistencies, duplicates. Report how decisions were made.
- **Normalization/Transformation**: Scaling, encoding categorical values, timestamp alignment, etc.
- **Feature Engineering**: Derive relevant features for nodes, edges, temporal dimensions (for GNN, this is crucial).
- **Graph Construction** (if applicable): Document how raw traffic data is converted into nodes/edges with features.
- **Documentation**: Keep a preprocessing log (what was removed, imputed, transformed).
- **Code & Reproducibility**: Preprocessing should be scripted (Python, R, etc.), not manual, with version control.

Typical Deliverable:

- Preprocessing pipeline scripts (clean, transform, feature extract).
- Summary statistics before and after preprocessing.
- Visuals: distribution plots, missing data maps, correlation heatmaps.

Best Practices (PhD-Level Work)

1. **Data Management Plan (DMP)** — required in many grants/dissertations. Includes storage, backup, access, sharing plan.
2. **Standardized File Formats** — use CSV, Parquet, JSON, or HDF5 instead of proprietary/fragile formats.
3. **Automated Pipelines** — preprocessing steps in notebooks/scripts, not Excel/manual editing.
4. **Version Control for Data & Code** — Git for scripts, DVC (Data Version Control) or similar for large datasets.
5. **Reproducibility Package** — code + data (or at least metadata + synthetic sample if data are private).
6. **Benchmark Alignment** — if using public datasets, preprocessing should be comparable with benchmarks in the literature.
7. **Validation Split** — document how data is partitioned (train/validation/test, time-based splits, cross-validation).

Evaluation Sheet for *Month 1–2 (Dataset Collection and Preprocessing)*

Metric	0	1	2	3	4	5	Score
Identification of Relevant Datasets(number & relevance)	None	1 dataset, weak relevance	1 dataset, moderate relevance	≥2 datasets relevant	≥2 datasets highly relevant + some novelty	≥3 datasets highly relevant, comprehensive coverage	
Documentation of Data Sources & Metadata (size, variables, license)	None	Minimal info	Basic info for some datasets	Basic info for all datasets	Complete info for all datasets	Complete + structured data sheets (versioning, access details)	
Ethics & Licensing Compliance (citations, usage rights, privacy)	None	Unclear status	Some compliance	Documented compliance for most	Documented compliance for all	Full compliance + IRB/ethics statements if needed	
Data Cleaning Quality (missing values, outliers, duplicates)	None	Ad hoc, undocumented	Basic cleaning for some variables	Documented cleaning for main variables	Documented cleaning for all variables + rationale	Automated, reproducible cleaning pipeline with rationale	
Preprocessing Pipeline Reproducibility(scripted pipeline)	None	Manual, undocumented	Some scripts but incomplete	Scripted pipeline, partial documentation	Scripted pipeline, documented, versioned	Fully automated pipeline, documented & version-controlled (e.g., Git/DVC)	
Feature Engineering & Suitability for Model (node/edge features)	None	Basic features, unclear link	Some features linked to problem	Relevant features derived	Well-justified features + graph construction documented	Innovative features + graph construction pipeline reproducible	
Data Readiness for Modeling (format, splits, transformations)	None	Raw data only	Some transformation	Fully transformed but no split	Fully transformed + documented split	Fully transformed + documented split + ready-to-run code	
Deliverables & Timeliness (scripts, data sheets, summary stats)	None	Very late/incomplete	Late/incomplete	On time but incomplete	On time, most deliverables	On time, complete deliverables incl. summary stats & visuals	

Core Objectives of *Conceptual Model Design & Framework Setup*

Objective	What It Means for the Student
Translate research gaps into a model concept	Show explicitly how your proposed model addresses identified gaps.
Define components and workflow	Inputs, processes, outputs, assumptions, boundaries.
Establish theoretical foundations	Anchor the model in existing theories or frameworks.
Plan evaluation	Define baseline models, metrics, and datasets for testing.
Create a reproducible environment	Prepare codebase, dependencies, data schema, and version control for later implementation.

Advisor's Expectation (Deliverables & Standards)

1. Clear Conceptual Model

- **Architecture Diagram**: A schematic of the model's main components (modules, data flow, input/output).
- **Component Description**: Written description of each module/layer and how it contributes to the goal.
- **Assumptions & Scope**: What is included/excluded? Limitations stated explicitly.
- **Theoretical Basis**: References to key papers/approaches that inspired the design.

Deliverable:
A **design document** with text (a clear narrative explaining how the model works) and diagrams (**architecture diagram** in a professional tool (LaTeX TikZ, draw.io, Visio, etc.).).

---

2. Framework Setup

- **Baseline Models**: Identify at least two relevant baselines for fair comparison. Justify selection.
- **Data Schema & Feature Definition**: Define exactly how raw data will be represented as input (e.g., for a GNN, how nodes/edges and features are structured).
- **Experimental Plan**: Preliminary choice of evaluation metrics and validation strategy.
- **Software Environment**: Set up and document libraries, versions, and hardware needs (e.g., PyTorch Geometric, DGL, CUDA versions).
- **Repository & Version Control**: Create a reproducible repository with README, environment files, and initial structure (folders for data, code, notebooks).

Deliverable:
A document & slide deck (**slides for the design review meeting**) describing design and setup. **Comparison table** (in the document & slides) showing how the proposed model differs from baselines. **Git repository** with a folder structure, environment files, and placeholders for modules (A working environment ready for implementation, even if code is just placeholders.)

Best Practices (PhD-Level)

- **Iterative Prototyping**: Start with a simplified version of the model; refine based on feedback.
- **Modularity**: Design components so they can be independently tested or replaced.
- **Transparency**: Document decisions (why you chose certain architectures, metrics, baselines).
- **Scalability**: Ensure the design can handle the full dataset; test on a subset early.
- **Reproducibility**: Environment files (requirements.txt, environment.yml, Dockerfile); versioned code.
- **Alignment With Evaluation**: Metrics, baselines, and data splits defined now, not after coding.

Evaluation Sheet for *Month 3 (Conceptual Model Design & Framework Setup)*

Metric	0	1	2	3	4	5	Score
Architecture Clarity(diagram & description of model components)	No diagram or unclear model	Sketchy diagram; no explanation	Partial diagram with major gaps	Complete diagram but vague explanation	Complete diagram with clear explanation of most modules	Complete, professional diagram + fully explained modules and data flow	
Component Justification (link to literature & research gaps)	No justification	Minimal references	Some references but not tied to gaps	Most modules justified by literature	All modules justified + gaps addressed	All modules justified with strong, critical discussion of gaps and novelty	
Baseline Models Selected & Justified	None	1 baseline without justification	1–2 baselines weakly justified	≥2 baselines, partly justified	≥2 baselines with solid justification	≥2 baselines with strong justification and clear evaluation plan	
Data Schema & Feature Plan (how raw data becomes input to model)	Not defined	Sketchy/unclear schema	Basic schema defined	Schema + some feature details	Schema + all features defined clearly	Schema + features + rationale for selection documented	
Evaluation Plan(metrics, validation, datasets)	None	Very vague plan	Basic metrics only	Metrics + dataset splits defined	Metrics + dataset splits + baselines defined	Complete evaluation strategy with justification	
Environment & Framework Setup(code repo, reproducibility)	None	Incomplete or missing files	Minimal folder structure	Structured repo + partial environment files	Structured repo + full environment file + version control	Fully reproducible repo (README, env files, version control, placeholders for modules)	
Deliverables Quality(design document / slides)	None	Fragmentary notes	Basic document	Document with diagrams	Well-written document/slides with diagrams	Professional, polished document/slides suitable for publication or defense	

Core Objectives of *First Phase Implementation*

Objective	What It Means for the Student
Code Implementation of Core Modules	Translate conceptual modules into working code (even if simplified). Each module should run independently and integrate into the pipeline.
Reproducibility & Modularity	Implementation should be modular, well-documented, version-controlled, and reproducible on another system.
Initial Data Integration	Load and process the datasets prepared in the preprocessing stage; verify the input-output workflow.
Baseline Integration	Include baseline methods (from the conceptual design) for initial comparisons.
Preliminary Evaluation	Compute initial metrics on a subset of data to verify functionality.
Documentation & Code Quality	Clear comments, README, docstrings, and coding standards followed.

Advisor's Expectation (Deliverables & Standards)

1. Implementation of Core Modules

- **Node Feature Enrichment**: Implement at least the first module(s) of the proposed GNN architecture.
- **Dynamic Link Analysis**: Implement preliminary version for a small subset of the network.
- **Pipeline Integration**: Modules should connect and produce output without errors.

Deliverables:

- **Code repository** with:
  - Core module implementations (Python scripts / Jupyter notebooks)
  - Environment files
  - Unit tests
  - Baseline modules
  - Sample runs demonstrating module functionality

---

2. Reproducibility & Environment

- **Version Control**: Git repository with meaningful commit messages.
- **Dependencies**: requirements.txt or environment.yml.
- **Documentation**: README describing how to run code, sample inputs/outputs.

Deliverables:

1. **Notebook or report** summarizing:
   - Implementation decisions
   - Preliminary results (tables/plots)
   - Issues encountered and solutions
2. **Documentation**:

    - README with instructions
    - Comments and docstrings in code

---

3. Data Handling

- Implement data loaders for datasets prepared earlier.
- Include preprocessing steps in code or call preprocessing scripts.
- Ensure modularity to handle multiple datasets easily.

---

4. Baseline Models

- Include initial versions of baseline methods for early comparison.
- Document differences from proposed model.
- Compute preliminary metrics (accuracy, AUC, etc.) on small datasets.

---

5. Preliminary Results & Testing

- Run small-scale experiments to verify:
    - Modules produce expected outputs.
    - Training loop works (loss decreases or metrics reasonable).
    - Baseline vs. proposed module can be compared.
- Prepare simple visualizations (loss curves, metric tables) for feedback.

---

6. Code Quality & Best Practices

- Modular code: each function/class has a single responsibility.
- Clear variable names, comments, and docstrings.
- Consistent style (PEP8 for Python).
- Avoid hard-coded paths; use config files or parameters.
- Scripts/notebooks must reproduce results on another system.

Evaluation Sheet for *Month 4 (First Phase Implementation)*

Metric	0	1	2	3	4	5	Score
Core Module Implementation(planned modules implemented)	None	Very few modules implemented	Some modules partially implemented	Main modules implemented	Training + evaluation loop	Running model with preliminary results	
Pipeline Functionality(end-to-end workflow)	None	Cannot run end-to-end	Runs partially on small data	Runs end-to-end on subset of data	Runs end-to-end on full small dataset	Runs end-to-end on full dataset without errors	
Baseline Models Integration	None	Baselines not included	One baseline implemented, partial	Baselines implemented, partial runs	Baselines implemented and run	Baselines fully implemented, run, and results compared with proposed modules	
Preliminary Evaluation(metrics, outputs)	None	No metrics	Metrics computed for 1 module	Metrics for multiple modules	Metrics computed + simple visualization	Metrics computed + visualizations + initial analysis	
Reproducibility(scripts, environment, version control)	None	No scripts or env, single commit	Scripts exist but not runnable, sparse commits	Scripts runnable, env partially documented, regular commits	Scripts + env fully documented, regular + clear messages	Fully reproducible repo with instructions, environment, and version control	
Code Quality & Documentation(modular, readable, commented)	None	Poorly structured, no comments	Partial comments, some structure	Modular + comments for most functions	Modular + well-commented + docstrings	Highly modular, fully commented, docstrings, follows coding standards	
Functionality on Dataset	None	Preprocessing only	Loads data	Trains on sample	Trains on full dataset	Trains + outputs preliminary metrics	
Deliverables (working repo, notebook/report)	None	Incomplete	Basic repo or notebook	Repo + notebook with partial results	Repo + notebook with results + brief analysis	Complete repo + notebook/report summarizing implementation, results, and issues	

Core Objectives of Testing, Refinement & Preliminary Results

Objective	What It Means for the Student
Evaluation Coverage	Test the model on multiple datasets (or multiple subsets of the dataset) to demonstrate generalizability and robustness.
Metric Reporting	Compute relevant metrics (F1, accuracy, AUC, precision/recall, link prediction accuracy, etc.) for each dataset.
Baseline Comparison	Compare performance against baselines or prior methods to demonstrate improvements or trade-offs.
Refinement & Tuning	Identify weaknesses, tune hyperparameters, and refine the model based on preliminary results.
Visualization & Interpretation	Present results in tables, plots, and graphs with meaningful interpretation.
Reproducibility	All experiments should be reproducible; document dataset splits, random seeds, environment.
Interim Reporting	Provide a clear report or notebook summarizing methods, datasets, metrics, results, and next steps.

Advisor's Expectation (Deliverables & Standards)

1. Evaluation Coverage

- Run the model on multiple datasets or at least multiple folds/splits.
- Ensure datasets reflect the diversity of conditions the model is expected to handle.
- Document exactly which datasets were used, preprocessing applied, and which subset was used for preliminary tests.

2. Metrics Reporting

- Choose metrics appropriate to the problem (e.g., F1, AUC, accuracy for classification/link prediction; MSE/RMSE for regression).
- Report metrics in **tables or charts**, not just text.
- Include standard deviations or confidence intervals when applicable.

3. Baseline Comparison

- Compare results with at least two baseline methods.
- Include **numerical tables and visual comparisons**.
- Discuss strengths and weaknesses relative to baselines.

4. Refinement & Hyperparameter Tuning

- Document attempts to improve the model (hyperparameters, features, architecture tweaks).
- Keep track of **which experiments were successful and why**.
- Avoid overfitting by using validation/test splits properly.

5. Visualization & Interpretation

- Use graphs, plots, heatmaps, or tables to make results clear.
- Highlight key findings (e.g., which nodes/edges most affect congestion in traffic networks).
- Provide **insightful commentary**, not just numbers.

6. Reproducibility

- Use fixed random seeds where needed.
- Document environment (Python version, library versions, GPU/CPU used).
- Make all scripts reproducible: training, evaluation, plotting.

7. Interim Reporting

- Draft a report or notebook including:
    - Background & objectives
    - Methods implemented
    - Preliminary results with tables/plots
    - Comparison with baselines
    - Limitations, next steps, and planned refinements

Deliverables

1. **Code repository** containing:
    - Scripts for evaluation on all datasets
    - Baseline models implementation
    - Hyperparameter tuning experiments
2. **Tables & plots** showing metrics and comparisons
3. **Notebook or report** summarizing:
    - Methods and datasets
    - Preliminary results
    - Comparison with baselines
    - Analysis and insights
    - Next steps for model refinement

Evaluation Sheet for *Month 5 (Testing, Refinement & Preliminary Results)*

Metric	0	1	2	3	4	5	Score
Evaluation Coverage (# datasets / subsets tested)	None	1 dataset, minimal testing	1 dataset, some testing	≥2 datasets/subsets, partial coverage	≥2 datasets/subsets, most covered	≥2 datasets/subsets, comprehensive coverage + robustness tests	
Metrics Reported(F1, AUC, link prediction accuracy, etc.)	None	Only 1 metric, incomplete	1–2 metrics, partial	2–3 metrics, reported	2–3 metrics, well-reported in tables/plots	≥3 metrics, well-reported with plots, tables, and interpretation	
Comparison with Baselines	None	Baselines missing	1 baseline, partial results	≥1 baseline, full results	≥2 baselines, comparison clear	≥2 baselines, comparison clear, interpreted, strengths & weaknesses discussed	
Refinement & Hyperparameter Tuning	None	Minimal tweaks, undocumented	Some tweaks, partial documentation	Tweaks documented for main modules	Tweaks documented + preliminary improvement observed	Systematic refinement, documented experiments, improved metrics demonstrated	
Visualization & Interpretation	None	Minimal visuals, no interpretation	Basic visuals, limited commentary	Clear visuals + basic interpretation	Clear visuals + detailed interpretation	Professional-quality visuals + insightful interpretation for all key findings	
Reproducibility(scripts, env, seeds)	None	Scripts exist but not runnable	Partial reproducibility	Scripts + environment documented	Fully reproducible pipeline	Fully reproducible pipeline + results verified on another system	
Interim Report / Notebook Quality (background, methods, preliminary results)	None	Fragmented notes	Basic report	Report with methods & results	Well-organized report/notebook with clear tables/plots	Professional, polished report/notebook suitable for advisor review or publication draft	