Data-Intelligent ANPR: A Scalable Framework for License Plate Recognition Under Real-World Data Constraints

Awiros

Abstract. This paper presents the design and evaluation of a production-grade Automatic Number Plate Recognition (ANPR) system built to handle both standard and non-standard Indian license plate formats under real-world deployment conditions. Accurate license plate recognition is an almost-solved problem if latency and cost are no object — frontier multimodal models like Gemini achieve strong accuracy on Indian plates with zero domain-specific training. The real constraint in production ANPR is throughput: real-time deployment cannot afford 6-second API round-trips or per-query cloud inference costs. The central challenge addressed here is therefore two-fold: a cold-start data problem — only 6,839 labeled samples from a publicly available dataset [3] were available at the start of development — and the need to produce a model that is both highly accurate and deployable within production latency constraints. We describe a data engineering pipeline spanning synthetic data synthesis, consensus pseudo-labeling, distribution-aware curation, VLM-assisted data cleanup, and state-balanced batch sampling — achieving a training corpus of 558,767 labeled samples. The resulting Awiros-ANPR-OCR, a 37M-parameter specialist model, achieves 98.42% accuracy while delivering sub-6ms on-device inference — a 1,260× latency advantage over Gemini — by investing in data engineering rather than model scale. The same base architecture scores 57.96% out-of-the-box; the gap to 98.42% is entirely a data story.

1. Introduction

Automatic Number Plate Recognition (ANPR) is a critical component of intelligent transportation systems (ITS), parking management, toll enforcement, and public safety monitoring. A functional ANPR system must accurately transcribe alphanumeric characters from license plates captured across a wide range of conditions: varying illumination, motion blur, partial occlusion, and diverse plate formats.

Deep learning approaches to OCR have largely replaced classical image processing pipelines, offering strong generalization when sufficient labeled training data is available. However, obtaining such data in domain-specific contexts is expensive and time-consuming. This scarcity is especially pronounced for Indian license plates, which present two broad categories of recognition challenge.

Standard plates conform to the IND-mandated format: a fixed two-character state code prefix, district and series identifiers, and a four-digit sequence, rendered in a prescribed font on a white or yellow background. **Non-standard plates** deviate from this template in one or more dimensions: dual-row layouts used on commercial vehicles, plates issued under schemes that do not follow the conventional state-code prefix structure, region-specific font variants, or plates exhibiting physical degradation, handwritten characters, or aftermarket modifications. Non-standard plates account for a disproportionate share of recognition failures in deployed systems, yet are systematically underrepresented in publicly available datasets.

This paper addresses the problem of high-accuracy ANPR under a severe data constraint: only 6,839 labeled samples from a publicly available dataset were available at the start of development, against an estimated requirement of 80,000+ for robust coverage across both standard and non-standard formats. Rather than treating this as a fundamental limitation, we treat it as a data engineering problem — designing a pipeline that constructs a high-quality, distribution-balanced corpus from a combination of synthetic generation, unlabeled field data, and targeted human annotation.

The primary contributions of this work are:

- A synthetic plate generation engine that renders photorealistic training data across Indian state formats, fonts, and degradation conditions
- A consensus pseudo-labeling framework using agreement between independently trained models as a reliable proxy for ground-truth labels
- A non-linear, distribution-aware dataset split strategy that ensures training robustness across rare and common plate states
- A selective annotation feedback loop that maximizes the marginal utility of each human-labeled sample
- A VLM-assisted data cleanup stage using Tencent HunyuanOCR to suppress hallucination and remove low-quality samples
- A state-balanced batch sampler that enforces uniform state-code representation within each training batch
- A negative sample training strategy using unreadable plates labeled with an abstention token, suppressing hallucination on degraded inputs
- An end-to-end non-standard plate architecture that deprecates brittle multi-stage pre-processing cascades

2. Background and Related Work

2.1 License Plate Recognition Systems

ANPR systems typically decompose into two stages: detection (locating the plate within the full image) and recognition (transcribing characters from the cropped plate). Modern approaches use deep neural networks for both stages, with architectures ranging from CNNs with CTC loss [4] to attention-based encoder-decoder models.

Indian license plates present specific challenges beyond what standard benchmarks capture. Standard IND-format plates are relatively tractable for modern OCR systems. The harder problem is non-standard plates: dual-row commercial vehicle layouts, plates deviating from the conventional state-code prefix structure, regional font variants, and plates exhibiting severe physical degradation. Prior deployed systems relying on multi-stage pipelines — separate segmentation, row splitting, and per-region recognition — are particularly fragile on non-standard formats, as errors at any stage compound irreversibly downstream. These factors make domain-specific training data and end-to-end architectures essential for robust performance across the full distribution.

2.2 Data-Centric AI

The data-centric AI paradigm reframes model development around improving data quality and coverage rather than model architecture. Key techniques include label noise correction, active learning, curriculum learning, and distribution-aware sampling. Effective data curation can often yield larger accuracy gains than architectural changes.

2.3 Synthetic Data for OCR

Synthetic data generation has a long history in text recognition. Datasets such as SynthText [5] and MJSynth [6] demonstrated that models trained on rendered text can achieve competitive performance on real-world benchmarks. For domain-specific tasks, custom rendering engines allow fine-grained control over format, font, and degradation distributions.

2.4 Semi-Supervised and Pseudo-Label Learning

Semi-supervised learning (SSL) methods leverage unlabeled data to improve model performance beyond what labeled data alone enables. Pseudo-labeling [7] — assigning model predictions as labels for unlabeled data and retraining — is one of the simplest and most widely used SSL techniques. FixMatch [8] extends this with consistency regularization, using data augmentation as a consistency constraint. Our consensus labeling approach adapts pseudo-labeling by requiring agreement between two independently trained models, substantially improving label reliability.

2.5 VLMs for Data Quality Control

Large vision-language models (VLMs) with strong zero-shot OCR capabilities can play a useful role in data pipeline design beyond serving as end-deployment systems. Rather than acting as annotation oracles, VLMs can function as quality filters: their outputs on a given sample provide a signal for detecting low-quality or hallucination-prone inputs — blank plates, heavily degraded images, or crops where a weaker model produces confidently wrong predictions. This heuristic use of VLMs for data cleanup is distinct from automated labeling and does not require VLM outputs to be ground-truth reliable.

3. System Design

3.1 Task Formulation

Given a cropped license plate image I, the model produces a character sequence $S = \{c_1, c_2, \ldots, c_n\}$ corresponding to the plate alphanumerics. For standard single-row plates, this is a standard sequence recognition problem. For non-standard dual-row plates, the model must learn to read top and bottom rows in the correct order and concatenate the result as a single normalized sequence.

3.2 Architecture Overview

The recognition model follows an encoder-decoder structure based on PP-OCRv5 [1]. The encoder extracts visual features from the plate image using a convolutional backbone. The decoder generates the character sequence using attention over encoder features, learning to attend to relevant spatial regions at each output step.

This end-to-end formulation is key to handling non-standard plate layouts robustly. Earlier multi-stage pipelines decomposed the recognition problem into sequential steps — perspective normalization, row segmentation, per-region character recognition, and sequence concatenation — each introducing its own failure modes. Errors at the segmentation stage propagated irreversibly through subsequent steps, making such pipelines particularly brittle on dual-row and other non-standard formats. The end-to-end architecture treats the full plate image as a single input and learns layout-specific reading order implicitly from training data, eliminating all intermediate failure points.

4. Dataset Overview

4.1 Row Type and Data Source Distribution

The final dataset contains 558,767 samples organized by plate layout. All dual-row plates originate from consensus pseudo-labeling of unlabeled field data (Section 5.2); all remaining

sources — including synthetic data and human-annotated rare-state plates — are single-row.

Table 1: Dataset Composition by Row Type

Row Type	Count	%
1-Row	456,993	81.8%
2-Row	101,774	18.2%
Total	**558,767**	**100%**

4.2 State-wise Distribution

The actual training pipeline uses consolidated label files: `train.txt` (423,834 samples) and `val.txt` (134,933 samples), totalling 558,767 labeled plate images across 83 unique state/UT code prefixes.

The top 5 states account for 89.3% of the dataset: KA (180,755; 32.3%), HR (180,158; 32.2%), RJ (76,493; 13.7%), DL (48,326; 8.6%), and PB (13,205; 2.4%). The long-tail nature of this distribution — with 41 state codes having fewer than 10 samples — motivates the distribution-aware split strategy (Section 5.3) and state-balanced batch sampling (Section 5.6).

Table 2: State-wise Sample Distribution (Top 10 States)

State Code	Count	%
KA (Karnataka)	180,755	32.3%
HR (Haryana)	180,158	32.2%
RJ (Rajasthan)	76,493	13.7%
DL (Delhi)	48,326	8.6%
PB (Punjab)	13,205	2.4%
UP (Uttar Pradesh)	12,353	2.2%
MH (Maharashtra)	6,384	1.1%
GJ (Gujarat)	5,529	1.0%
NL (Nagaland)	3,793	0.7%
CH (Chandigarh)	3,691	0.7%
Top 10 Total	**530,687**	**95.0%**

5. Data Engineering Methodology

The core contribution of this work is methodological: a data engineering pipeline that constructs a high-quality, distribution-balanced training set from a combination of publicly available labeled data, synthetic generation, unlabeled deployment data, VLM-assisted quality filtering, and targeted human annotation.

5.1 Seed Data and Synthetic Bootstrapping

Development began with 6,839 labeled samples from a publicly available Indian vehicle dataset [3]. This seed corpus, while insufficient for robust deployment-grade training on its own, provided a real-world foundation to anchor the synthetic generation process and bootstrap the initial model.

To bridge the gap toward the estimated 80,000+ samples needed, a domain-specific synthetic plate generator was developed. The engine renders plates with controlled variation across: all Indian state and union territory formats; background textures simulating reflective sheeting, aging, and dirt; geometric distortions including perspective warp, affine skew, and barrel distortion; and photometric degradations such as motion blur, Gaussian noise, JPEG artifacts, and shadow gradients.

The combined corpus of real seed data and synthetic plates was used to train a seed model (Model B). This model served two purposes: it enabled immediate experimentation while the unlabeled field data pipeline was being established, and it became one of the two models used in the consensus pseudo-labeling stage.

5.2 Consensus Pseudo-Labeling

To scale the training set using unlabeled field data without manual annotation, a consensus labeling framework was designed. Plate crops were extracted from a mixed corpus of live deployment footage and internal CCTV archives using a detection model with a dedicated license plate class. The pool exhibits a median resolution of 142×49px (mean 151×55px) and a median aspect ratio of 3.18, consistent with standard landscape plate layouts. The wide spread — resolutions ranging from 19×8px to 720×306px and aspect ratios from 0.59 to 5.19 — reflects the full range of real-world CCTV capture conditions, from high-resolution close-range frames to small, distant, or perspective-distorted detections. These unlabeled crops were then processed through two independently trained models:

- **Model A:** Trained on the publicly available seed dataset, representing one training distribution
- **Model B:** The synthetic-bootstrapped seed model described above, trained on a different data distribution

The labeling heuristic relies on a probabilistic argument: two models trained on independent datasets are unlikely to produce the same incorrect transcription. Formally, if errors are approximately independent:

$$P(M_A{=}M_B{=}\text{err}) \approx P(M_A\,\text{err}) \cdot P(M_B\,\text{err} \mid \text{same}) \ll P(M_A\,\text{err}) \tag{1}$$

Images where both models produced identical outputs were auto-labeled with the consensus transcription and added to the training pool. While it is theoretically possible for both models to agree on the same incorrect output — for instance, on systematically ambiguous plate characters or heavily degraded inputs — such cases are expected to be rare relative to the volume of correctly agreed samples. The signal-to-noise ratio of the consensus corpus is therefore substantially higher than that of single-model pseudo-labeling, where any individual model error passes through unchecked.

5.3 Distribution-Aware Dataset Curation

Field deployment data is heavily skewed toward high-traffic states. A non-linear bucket-wise split was applied:

- **High-volume states** (top quartile): 35% validation / 65% training
- **Medium-volume states**: linearly interpolated split
- **Rare states** (bottom quartile): 5% validation / 95% training

This strategy ensures the validation set is a sensitive benchmark for rare-state generalization, while the training set provides maximum coverage of edge cases.

5.4 Selective Annotation Feedback Flywheel

As model quality improved across training iterations, samples were prioritized for human annotation based on: state rarity (underrepresented states flagged as highest priority), model uncertainty (low character-level confidence), and prediction entropy (high-entropy beam search outputs). This active learning–inspired approach ensures every human-labeled sample addresses a genuine distribution gap.

5.5 VLM-Assisted Data Cleanup

A practical failure mode in OCR model training is the presence of low-quality samples that drive hallucination: blank plates, heavily blurred or overexposed images, and crops where the detection model returned a non-plate region. Models trained on such samples learn to produce confident but incorrect outputs on degraded inputs, rather than abstaining.

Tencent HunyuanOCR [2] was used heuristically as a quality filter to identify and remove these samples. Since HunyuanOCR is a strong general-purpose OCR system, samples on which it produced no output or an implausibly short string were flagged as likely unreadable and removed from the training pool. This cleanup stage did not use HunyuanOCR outputs as ground-truth labels; it used the presence or absence of a plausible output as a proxy for image quality, controlling the hallucination surface of the training data.

5.6 State-Balanced Batch Sampling

To prevent training dynamics from being dominated by high-frequency states, the dataloader was modified to enforce state-balanced batch composition. Samples were grouped by state code (the first two characters of each label). Within each batch, states were sampled uniformly before sampling individual plates within each state. This ensures that rare states receive gradient signal proportional to their importance rather than their frequency.

5.7 Negative Samples and Unreadable Plate Handling

Despite the VLM-assisted cleanup stage, a residual set of genuinely unreadable plates — heavily blurred, occluded, or physically damaged — remains in the training pool by design. These are labeled with an abstention token `<UNK>` and included in training. This regularizes the model's confidence distribution on degraded inputs, teaching it to output the abstention token instead of a confabulated sequence, and complements the upstream cleanup by handling edge cases the quality filter may not catch.

6. Experimental Evaluation

6.1 Validation Dataset

All systems were evaluated on a shared held-out validation set constructed using the distribution-aware split described in Section 5.3. The validation set covers plates from across all Indian state codes, with the non-linear split ensuring sensitivity to both common-case accuracy and rare-state generalization, including both standard and non-standard plate formats.

6.2 Compared Systems

We evaluate five systems on the same validation set:

- **Awiros-ANPR-OCR (Ours):** End-to-end architecture trained with the full data pipeline described in this paper. The model is based on the PP-OCRv5 [1] encoder-decoder backbone, fine-tuned on our curated 558,767-sample corpus with domain-specific optimizations for Indian license plates
- **PP-OCRv5 Pretrained:** The upstream PP-OCRv5 server recognition model [1] evaluated with its original pretrained weights and full multilingual dictionary (18,383 characters), without any fine-tuning on Indian license plate data. This baseline isolates the contribution of our data engineering pipeline
- **Google Gemini 3 Flash (gemini-3-flash-preview):** General-purpose multimodal LLM evaluated via the Gemini API with a standardized transcription prompt
- **Google Gemini 2.5 Flash (gemini-2.5-flash):** Previous-generation Gemini Flash model evaluated under identical conditions
- **Tencent HunyuanOCR:** Production-grade commercial OCR API evaluated via the Tencent Cloud API

All models receive identical pre-cropped plate images. No fine-tuning or domain adaptation was performed for the commercial baselines or the PP-OCRv5 pretrained model. Outputs were post-processed uniformly: whitespace stripped, characters uppercased.

6.3 Results

6.4 Discussion

The PP-OCRv5 pretrained baseline is particularly instructive: despite sharing the same model architecture as Awiros-ANPR-OCR, its accuracy drops from 98.42% to 57.96% overall and from 96.91% to 0.24% on 2-row plates. This demonstrates

Table 3: Recognition Accuracy, Latency, and Throughput on Held-Out Validation Set

System	Params	Overall Acc.	1-Row Acc.	2-Row Acc.	Latency Avg (ms)	Latency P50 (ms)	Throughput (img/s)
Awiros-ANPR-OCR (Ours)	37.3M	**98.42%**	**98.83%**	**96.91%**	**5.09**	**4.93**	**196.5**
Google Gemini-3-flash-preview	∼5-10B†	93.89%	94.70%	91.20%	6,430†	4,343†	0.2†
Google Gemini-2.5-flash	∼5B†	87.23%	89.66%	78.38%	—†	—†	—†
Tencent HunyuanOCR	996M	67.62%	76.65%	34.78%	309.15	300.70	3.2
PP-OCRv5 Pretrained	53.6M	57.96%	73.55%	0.24%	5.25	5.19	190.6

Latency measured on a single NVIDIA RTX 3090 GPU (batch size 1). †Gemini latency is end-to-end API round-trip (includes network, image upload, and server-side inference); parameter count is not publicly disclosed. Gemini 2.5 Flash latency was not separately benchmarked (—). Awiros-ANPR-OCR and PP-OCRv5 Pretrained use Paddle Inference; Tencent HunyuanOCR uses PyTorch with bfloat16.

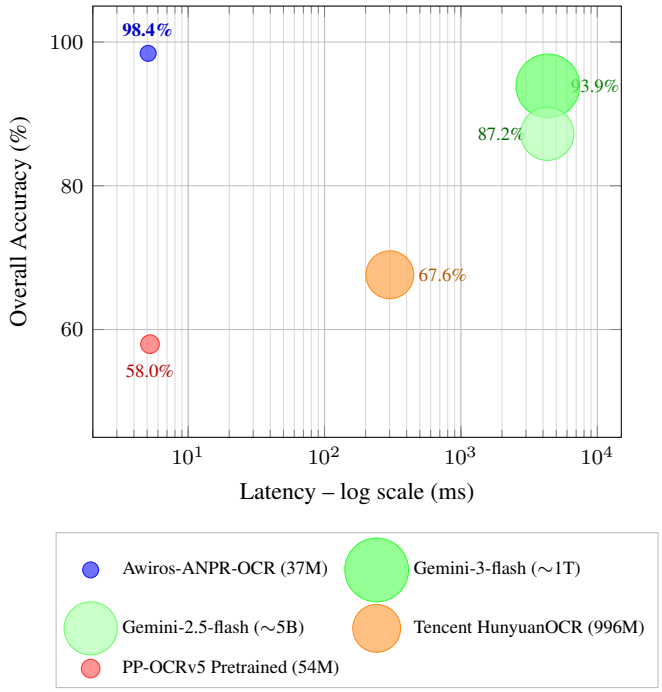


Figure 1: Accuracy vs. latency (log scale) for all evaluated systems. Circle area is proportional to parameter count. Awiros-ANPR-OCR achieves the highest accuracy at the lowest latency. Gemini latencies reflect API round-trip time including network overhead; Gemini 2.5 Flash latency is estimated as comparable to Gemini 3 Flash.

that the data engineering pipeline — not the model architecture — is the primary driver of performance on Indian license plates. The pretrained model, trained on a general multilingual corpus of 18,383 character classes, lacks the domain-specific knowledge needed for Indian plate formats and occasionally produces Chinese character predictions on degraded inputs.

The two Gemini baselines together illustrate both the capability ceiling of general-purpose frontier models and the gap that remains. Gemini 3 Flash achieves 93.89% overall — a strong result for a model with no domain-specific training — but drops to 91.20% on non-standard dual-row plates, where the lack of format-specific supervision is most exposed. Gemini 2.5 Flash trails further at 87.23% overall and 78.38% on 2-row plates, suggesting that even across model generations, the domain gap is not closed by scale alone. Neither model is deployable for real-time use given API round-trip latencies in the range of several seconds.

Awiros-ANPR-OCR, trained with distribution-aware curation targeting both standard and non-standard Indian plate formats, is specifically robust to these failure modes. The end-to-end architecture additionally eliminates the cascade of segmentation errors that degraded multi-stage pipeline performance on commercial vehicle plates.

6.5 Qualitative Comparison

Table 4 shows representative samples where Awiros-ANPR-OCR correctly transcribes the plate while all baselines produce errors. Common failure modes include confusing visually similar characters (Q→0, V→Y, M→R, B→8) and truncating dual-row plates. Notably, both Gemini versions share many of the same character confusion patterns, confirming that these errors reflect domain-specific visual ambiguities that general pre-training does not resolve.

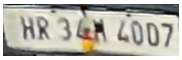
7. Deployment and Efficiency

7.1 Computational Footprint

The end-to-end architecture directly reduces per-inference CPU load relative to multi-stage pipelines. Earlier approaches executed multiple sequential image transformation operations — perspective correction, connected-component analysis, row segmentation — each contributing measurable latency and memory overhead. The new end-to-end model replaces all pre-processing with a single forward pass.

The model is exported to ONNX format and served via ONNX Runtime, enabling cross-platform deployment without deep learning framework dependencies. TensorRT conversion is supported for NVIDIA edge hardware, providing further latency reduction on constrained deployment targets.

Table 4: Qualitative Comparison on Selected Samples — Awiros correct, all baselines incorrect

Plate Image	Ground Truth	Awiros (Ours)	Gemini 3	Gemini 2.5	Tencent
	UP14BQ9208	UP14BQ9208	UP14B09208	UP14B09208	UP14B
	HR35M2576	HR35M2576	HR35R2576	HR35R2576	HR35K2576
	HR34M4007	HR34M4007	HR34H4007	HR34M40D7	HR36M4007
	HR51BV8223	HR51BV8223	HR51BY8223	HR51BY8223	HR51BY8223
	HR38AB2421	HR38AB2421	HR38A8242	HR38A82421	HR38A
	HR12AX8522	HR12AX8522	HR12AX0522	HR12AX0522	HR12AX0522
	HR46E0227	HR46E0227	HR26E0227	HR26E0227	HR6E0227

Incorrect predictions are shown in red. Recurring character confusions: Q→0, M→R/K/H, V→Y, B→8, 8→0, 4→2. Tencent additionally truncates dual-row and low-contrast plates.

7.2 Continuous Improvement via Ongoing Pseudo-Labeling

The consensus pseudo-labeling and feedback flywheel components are designed for ongoing operation beyond the initial training cycle. As the deployed model generates predictions on live data, high-confidence consensus samples — where the production model and a validation reference agree — are automatically added to the growing training pool. This enables progressive accuracy improvement without additional annotation cost.

8. Limitations and Future Work

The consensus labeling framework depends on the availability of a second independent model trained on a different data distribution. In a true cold-start scenario with no separately trained model available, only the synthetic-bootstrapped model is available for pseudo-labeling, reducing the diversity of the consensus signal. Future work will investigate single-model confidence calibration as a fallback pseudo-labeling signal.

The synthetic data generator currently targets Indian license plate formats. Extension to international formats (European, Middle Eastern, Southeast Asian) would require format-specific rendering templates and is left for future work.

The distribution-aware split strategy uses state-level bucketing as a proxy for distribution coverage. More granular axes — plate age, physical condition, capture angle, and time-of-day — could further improve validation sensitivity and training robustness. Non-standard plate formats in particular may benefit from explicit sub-category stratification during both curation and evaluation.

Future directions include: online continual learning to reduce the retraining cycle, improved pseudo-label confidence calibration using temperature scaling, and extension to video-based ANPR where temporal consistency can be exploited as an additional signal.

9. Conclusion

This paper presented a data-intelligent ANPR framework that achieves high recognition accuracy without reliance on large manually annotated datasets. Starting from 6,839 publicly available labeled samples, we grew the training corpus to 558,767 samples by combining synthetic data synthesis, consensus-based pseudo-labeling on field-collected plate crops, distribution-aware curation, VLM-assisted data cleanup, state-balanced batch sampling, and negative sample training. The resulting model handles both standard and non-standard Indian license plate formats end-to-end, eliminating the brittle multi-stage pre-processing pipelines that prior approaches relied upon.

The broader takeaway is not that large models are ineffective

— Gemini 3 Flash's zero-shot 93.89% accuracy on a domain it was never trained for is genuinely impressive, and the consistent gap between Gemini generations on non-standard plates confirms that this is a domain alignment problem, not a scale problem. The takeaway is that production deployment imposes constraints that make scale-based solutions non-viable regardless of accuracy: API round-trips in the range of several seconds preclude real-time enforcement, and per-query cloud costs are unsustainable at production scale. The evidence is in the architecture comparison: PP-OCRv5 pretrained and Awiros-ANPR-OCR share the same backbone — one scores 57.96%, the other 98.42%. The model is not the variable. The core contribution is methodological: a data pipeline generalizable to any domain-specific OCR task where annotation cost is prohibitive, and where the deployment reality demands both accuracy and efficiency.

References

[1] PaddlePaddle Authors, "PP-OCRv5: A server-grade multilingual OCR system," PaddleOCR, GitHub, 2025. Available: `https://github.com/PaddlePaddle/PaddleOCR`

[2] Tencent Cloud, "Tencent HunyuanOCR — General Text Recognition API," Tencent Cloud Documentation, 2025. Available: `https://cloud.tencent.com/product/ocr`

[3] S. Saisirishan, "Indian Vehicle Dataset," Kaggle, 2020. Available: `https://www.kaggle.com/datasets/saisirishan/indian-vehicle-dataset`

[4] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 11, pp. 2298–2304, 2017.

[5] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proc. IEEE CVPR*, 2016, pp. 2315–2324.

[6] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Synthetic data and artificial neural networks for natural scene text recognition," in *NIPS Workshop on Deep Learning*, 2014.

[7] D.-H. Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *ICML Workshop on Challenges in Representation Learning*, 2013.

[8] K. Sohn, D. Berthelot, C.-L. Li *et al.*, "FixMatch: Simplifying semi-supervised learning with consistency and confidence," in *Proc. NeurIPS*, 2020.

A. Evaluation Setup

All models were evaluated under identical input conditions. Plate images were pre-cropped to the plate region using the same detection model before being passed to each recognition system, ensuring fair comparison at the recognition stage only.

Commercial API calls to Google Gemini (gemini-3-flash-preview, gemini-2.5-flash) and Tencent HunyuanOCR used the following standardized prompt: *"Transcribe the license plate number in this image. Return only the alphanumeric characters with no spaces, hyphens, or special characters."* All outputs were post-processed by stripping whitespace and converting to uppercase.

B. Synthetic Data Generation Details

The plate renderer supports the following parameterized variation axes:

- **Plate format:** All Indian state and union territory codes with correct font-face and character spacing conventions, including non-standard dual-row layouts
- **Geometric transforms:** Perspective ($\pm 15°$), affine skew ($\pm 10°$), barrel distortion ($k = 0.0$–0.3)
- **Photometric:** Motion blur (kernel 3–15px), Gaussian noise ($\sigma = 5$–25), JPEG compression (quality 40–95), brightness jitter ($\pm 30\%$), shadow gradients
- **Plate conditions:** Clean, weathered (color fade), dirty (random occlusion patches), damaged (character region masking)

Model Submission Options

Users may submit models in one of the following formats:

- **ONNX format:** A standalone `.onnx` file exported with fixed input layout and consistent tensor naming.
- **Binary deployment format:** Fully exported inference-ready binaries (e.g., OpenVINO `.xml` + `.bin` pair) generated from the same model checkpoint.

For ONNX submissions, the model must preserve the expected OCR input layout (typically `NCHW`, e.g., `1x3x48xW` with dynamic or fixed width) and consistent precision (FP32 or FP16).

For binary deployments, both files must originate from the exact same export step, ensuring matching tensor names, shapes, and precision.

Evaluation Pipeline

Once submitted, the model undergoes the following benchmarking procedure:

1. Model integrity and runtime compatibility validation.

2. Inference on the standardized validation manifest (`val.txt` in `image_path<TAB>label` format).
3. Generation of a structured JSON benchmark report.

The evaluation report includes:

- Throughput (samples/sec)
- Latency statistics (mean, P50, P90, P95)
- Processed vs failed inference counts
- OCR accuracy metrics (when decoding is enabled)

Dedicated Upload Interface

Users may either:

- Upload models directly via the Hugging Face evaluation interface, or
- Share deployment binaries with us for internal benchmarking.

Detailed submission guidelines, input format specifications, and evaluation constraints are documented in the project repository homepage.