**Related Work — RNNs for Exoplanet/Light-Curve Analysis**

Josh Manchester

This section reviews how recurrent neural networks (RNNs) have been applied to space-telescope light curves and what that means for my plan. I focus on three works: (1) an LSTM-based flare detector that operates on Kepler and TESS data (Vida et al., 2021), (2) an ESN–autoencoder representation learner designed for Kepler time series (Kügler, Gianniotis, & Polsterer, 2016), and (3) a recurrent marked temporal point process (RMTPP) model that shows how an RNN can encode event timing history (Du et al., 2016). Together these papers explain why sequence-aware models matter, which RNN variants tended to work well in practice, and how basic timing signals can help separate real periodic events from look‑alike variability.

RNNs for light-curve event detection (Vida et al., 2021). Vida and colleagues trained and evaluated RNNs for detecting stellar flares in Kepler and TESS photometry. Although flares are brightenings and transits are dimmings, both problems require learning patterns in long, noisy sequences. Their best-performing network stacked several long short-term memory layers (LSTMs), used dropout for regularization, and applied a one‑unit sigmoid output for binary classification (Vida et al., 2021). The authors addressed class imbalance directly by weighting the minority class and tuned hyperparameters systematically. An important practical insight was that including “hard negatives” (non‑flare astrophysical signals) during training reduced false positives. Most relevant for our project, a model trained on Kepler generalized well to TESS even though the cadence and noise profiles differ. That result supports my plan to keep preprocessing consistent across sources, use stacked LSTMs or GRUs first, and report precision, recall, F1, and ROC–AUC in a way that is robust to threshold choice (Vida et al., 2021).

Two details from Vida et al. (2021) feed directly into my setup. First, the authors compared gated recurrent unit (GRU) and LSTM variants and found that stacked LSTMs handled “astrophysical noise” better in their flare task. I will therefore start with a 2–3 layer LSTM (128–256 units) and keep a GRU as a secondary baseline. Second, they used strong, simple regularization techniques—dropout and early stopping—and balanced the classes explicitly. Because transit positives will be rare in my small subset (≈10 known systems), I will follow the same approach and also try focal loss as a variant when class weighting is not enough to control false positives (Vida et al., 2021).

Sequence-aware representation with ESN–autoencoders (Kügler et al., 2016). While Vida et al. focus on supervised detection, Kügler and colleagues show how to learn lower‑dimensional representations of Kepler light curves that still respect sequence structure. They pair an Echo State Network (ESN) with an autoencoder: the ESN maps each light curve to a set of readout weights, and the autoencoder compresses and reconstructs those weights. Their key twist is to evaluate reconstructions in the original sequence space by pushing reconstructed weights back through the ESN to get a reconstructed light curve; the loss is computed on that sequence rather than only on the readout vector (Kügler et al., 2016). This simple but important change forces the model to preserve time‑dependent behavior.

For my work, the ESN–autoencoder paper provides two practical lessons. First, diagnostics should look at the whole sequence, not just a final embedding. I will therefore compute evaluation metrics on windows and on full sequences wherever possible and include plots that compare original and model‑highlighted time spans. Second, if I include any representation learning step (e.g., pre‑training or a feature extractor), I should prefer objectives that score performance in sequence space. Even without actually building an ESN–autoencoder, aligning training and evaluation with sequence fidelity helps avoid shortcut features and improves interpretability in error analysis (Kügler et al., 2016).

Timing-aware RNNs via RMTPP (Du et al., 2016). Du and colleagues introduce RMTPP, which uses an RNN to encode event histories and parameterize a temporal point process. Rather than predicting only labels, the model also learns when the next event is likely to occur based on the sequence so far. In other words, RMTPP turns the RNN into a history‑dependent intensity model that jointly handles event timing and event type (Du et al., 2016). Although RMTPP was demonstrated on domains like finance and healthcare, the idea is directly useful for periodic transit detection: a light curve with a planet has a regular rhythm (ingress to egress, repeat), while many false positives are aperiodic or quasi‑periodic. A light touch of timing information can help the classifier.

I will adapt RMTPP’s spirit in a lightweight way. I am not replacing the classifier with a full point‑process model, but I will add a simple timing helper that captures intervals between candidate transit edges or encodes a rough period guess when available. This helper can be an auxiliary feature or a small auxiliary loss that encourages consistency with a periodic template. The goal is to help the RNN internalize the idea that true signals repeat on a schedule, which reduces confusion with isolated dips or irregular stellar variability (Du et al., 2016).

Putting the three strands together. Vida et al. (2021) show that stacked LSTMs trained with class weighting and clean preprocessing can detect rare events in long, noisy sequences and even transfer across missions; I will mirror that recipe. Kügler et al. (2016) remind us to respect full‑sequence structure when learning representations or doing diagnostics; I will evaluate on sequences/windows and visualize model attention over time. Du et al. (2016) provide the blueprint for bringing timing into the loop; I will add a light timing signal to nudge the RNN toward periodicity without complicating the pipeline. These choices are simple, reproducible, and fit the scope of a small, curated dataset while still setting up a fair comparison against the team’s CNN/Transformer models.

Planned evaluation and error analysis based on related work. Following Vida et al. (2021), I will report precision, recall, F1, and ROC–AUC and examine confusion matrices for common failure modes. False positives will be grouped by likely source (e.g., variable stars, systematics, residual trends), and false negatives will be checked for very shallow or short transits. From Kügler et al. (2016), I will borrow the idea of sequence‑level diagnostics by plotting model scores across time to ensure the network’s focus lines up with actual transit windows. From Du et al. (2016), I will evaluate whether the timing helper reduces aperiodic false positives without hurting recall on genuine periodic signals. Together, these checks turn the related work into concrete, testable steps.

Implications for dataset size and generalization. Vida et al. (2021) showed that consistent preprocessing and balanced training data matter more than exotic architectures for robust performance; that is good news for a small curated subset. If results are promising, I can scale to more stars or additional sectors/quarters and repeat the same evaluation plan. When we expand, I will track whether the timing helper still helps, and whether stacked LSTMs remain better than GRUs, echoing the comparisons in Vida et al. (2021). Throughout, I will keep sequence‑based diagnostics from Kügler et al. (2016) so that model changes are grounded in observable time‑series behavior.

Summary. The RNN literature relevant to light curves argues for (a) stacked LSTMs or GRUs with explicit class‑imbalance handling and conservative regularization (Vida et al., 2021), (b) representation and diagnostics that respect sequence structure (Kügler et al., 2016), and (c) simple timing cues that help the network prefer periodic, transit‑like patterns (Du et al., 2016). I will carry these into my RNN baseline so that it is both technically sound and scoped for our project timeline.

# References (APA 6th)

Du, N., Dai, H., Trivedi, R., Upadhyay, U., Gomez‑Rodriguez, M., & Song, L. (2016). Recurrent marked temporal point processes: Embedding event history to vector. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 1555–1564). New York, NY: ACM.

Kügler, S. D., Gianniotis, N., & Polsterer, K. L. (2016). An explorative approach for inspecting Kepler data. Monthly Notices of the Royal Astronomical Society, 455(4), 4399–4405.

Vida, K., Bódi, A., Szklenár, T., & Seli, B. (2021). Finding flares in Kepler and TESS data with recurrent deep neural networks. Astronomy & Astrophysics, 652, A107.