**SLIDE 1**

Hi Everyone, I’m vidy and my project was on sequence based prediction of solar flares.

**SLIDE 2**

So before diving into my project, I wanted to give a little background into what solar flares actually are. A solar flare is a sudden release of magnetic energy in the Sun’s atmosphere which is triggered when magnetic field lines in the sun get twisted and then reconnect. This reconnection is what accelerates particles to near-light speed and emit radiation across the solar system that we detect. The energy involved in these are massive and equivalenet to billions of nuclear bombs going off. Solar flares are classified as A,B,C,M, and X class events which increase in severity and how often they occur.

**SLIDE 3**

And so we care about those high class solar flares because they can caused a variety of issues from taking down entire power grids for extended periods of time, causing billions of dollars in satellite damages orbiting the earth, and finally causing GPS systems around the world to shut down or worse provide incorrect information.

**SLIDE 4**

There are currently a few prediction models to predict the severity of solar weather as well as when they occur. The first is a physics based model called WSA Enlil hosted by NOAA, the national oceanic and atmospheric administration. This model takes in coronagraph imagery along with solar wind measurement from a Lagrange point near the Sun to predict when the Earth can expect peak solar storms. However, this model cant actually predict when solar flares will appear and has large lag times due to telemetry delay. Other models include a Solar flare classification model which uses 25 magnetic field parameters along with a magnetic imager to classify the types of flares. However this approach doesn’t provide a pre-flare classification. Another approach was to use a convolutional neural network to forecast based on image patches of active regions. My approach attempts to address some of the shortcoming by using a transformer encoder to ingest a continuous 48-hour history of 5-minut solar-wind parameters alongside daily solar-activity indicies such as sunspot numbers and Ap/Kp levels. By letting the model learn its own multi-scla features, I can capture both rpaid fluctuations and slow solar cycle trends without much manual feature engineering. And so the result of my approach is a 24-hour advance warning of flares that are an M-class or higher, as those ones pose potential danger.

**SLIDE 5**

So my project fuses together 3 different datasets that capture the slow solar-cycle trends and the more rapid drivers.

**SLIDE 6**

So my first dataset provides the long term geomagnetic context of the sun that solar wind data alone cannot supply. I use a daily Kp index reflects the impact of solar activity on a logarithmic scale of 1 to 9, think of it like the Richter scale for earthquakes but for geomagnetic storms. It also includes the sunspot number, which is more or less the number of sunspots we see and the F10.7 cm flux which is the measure of the suns radio emissions at that frequency. These fields are taken once daily from 1932 to present day, and during my eda, you’ll see that I dorward filled these daily fields onto my 5 minute grid to align with the other datasets.

**SLIDE 7**

The second dataset I used is the RHESSI flare catalogue which is an event based dataset of all solar flares observed between February 2002 and April 2018 which is over 100000 events. In this dataset each flare has a start, peak, and end time. It also includes the counts per second which referes to the number of photons detected by a sensor and is used to measure how active a solar flare is along with its energy range. It finally included X/Y positioning of these flares along the solar disk. Because I was using sequential data modeling, to turn these events into a target for my time series model, I created a binary “flare-flag” on the same 5 minute grid as the solar-wind data, setting it to 1 if any flare starts in that interval. After doing this it turned out that only around 0.3% of samples were positive so I made sure to use class weighting in my transformer’s loss to ensure that the model learned from the rare flare windows. This labelled series is basically the ground truth around which my entire pipeline revolves.  
**SLIDE 8**  
Finally the Solar wind dataset supplies 5 minute measurements of the planetary environment from the year 2000 onwards which is over 10 million rows. It first includes the north-south component of the interplanetary magnetic field (Bz). The next several fields all have to do with earths solar wind activity from speed of particles to proton density. Now you might be wondering why I chose to integrate downstream earth data as a response to the upstream solar flare. This is because they provide the global space-weather state and provide context on when solar regions are most likely to produce flares.

**SLIDE 9**

In my analysis of baseline models, I uncovered two critical issues. The first was year leakage. When I trained a Random Forest model on the raw merged hourly features, it scored an ROC-AUC of 0.642 and PR-AUC of 0.860 on the 2015-2017 validation period but then collapsed to ROC-AUC of 0.428 and PR-AUC of 0.005 on the 2018 test set. It turns out the model was simply using the absolute year as a proxy for solar-cycle phase, so it “knew” flare rates were high in cycle peaks and low in quiet years rather than learning true physical precursors. To address this, I removed all absolute date fields and replaced them with day of year sin/cos encodings, capturing seasonality without leaking solar-cycle information. The second issue was class imbalance. Only around 3% of all 5-minute windows contained an M or X class flare within the next 24 hours. Without correction, the model could trivially predict “no flare” and achieve high accurate. To address this, I applied class weights in the loss function and shifted the focus to PR-AUC rather than plain accuracy.

Slide 10

Before moving into the deep sequence models, I built three tiers of baselines to benchmark the task. Naïve is the trivial, ‘always-no-flare’ predictor and set the floor. Logistic Regression gives me a linear reference, posing the question if a simple weighted sum of features separate flares? It lifts the PR-AUC to 0.013 on test so there’s a faint linear signal but not even close for useful alerting. Next were Random Forest and Gradient Boosting but they too collapsed on the test window, which is actually what exposed my year leakage issue. From this I got a couple take away, that we need sequence context to generalize and that our deep models must be greatly better to show some semblance of real value.

Slide 12

So here is how every example is constructed. I slide a 48-hour window over the 5-minute grid on step at a time, which gives 576 timesteps per sample. Each timestep then carries 10 features, 6 solar wind variables and 3 daily geomagnetic solar indices that are forward filled. The label is a single bit: does an M-class or larger flare begin in the 24 hours immediately after this window?

Slide 13  
So the input window I just described goes into 2 bidirectional LSTM layers – one reading the sequence forward, and on backward with 64 units on the first and 32 on the second. Their output feed a small fully connected head that outputs a single probability of a flare starting in the following 24 hours. On the validation, the model maintains a high PR-AUC of around 0.86 with a ROC-AUC of near 0.6. Most importantly on completely unseen 2017-2018 test data, it achieves ROC-AUC of 0.58 and a pr-auc of around 0.1, which though it doesn’t seem like a lot is nearly a 20x improvement on the baseline models showing that modeling sequentially drives 24-hour forecast skill.

* NOTE TO SELF:  
  **Units (LSTM):** the number of memory cells per time‑step (e.g. 64 then 32), controlling the dimensionality of the hidden state; I used 64→32 units to balance capacity and over‑fitting.
* **Dense layer:** a fully‑connected neural layer where each input connects to every output node; I used Dense(32, ReLU) to learn nonlinear feature combinations before the final output.
* **Sigmoid output:** an activation σ(x)=1/(1+e^{-x}) that squashes values to [0,1], interpreted as a probability; I applied it in the final Dense(1, sigmoid) to predict flare probability.
* **ReLU:** the “Rectified Linear Unit” activation \max(0,x) that speeds training and avoids vanishing gradients; I used it in the hidden dense layer for efficient nonlinear transformation.
* **Patience (early stopping):** the number of epochs without improvement on the monitored metric before halting training; I set it to 5 epochs on validation PR‑AUC to prevent over‑training.

Slide 14

To squeeze extra skill from longer range interactions, I swapped the LSTM for a lightweight Transformer encoder. Each 48-hour sequence is project into a dimensional space, then pass through 2 self-attention blocks. And so, after global average pooling, it delivers the 24-hour flare probability. The transformer edges out the LSTM on the 2017-2018 test split reaching an ROC-AUC of 0.592 and PRAUC of 0.107 a roughly 6% precision recall gain.

Slide 15  
And so here is all the results that I previously described. Th blue line on the graph shows the Transformer on the 2017 – 2018 test, and you can see a slow increase in PR AUC model by model.

Slide 16

Here I just wanted to look back and see which input drive the Transformer’s decisions to ensure that the model is truly learning on physical precursors rather than the datetime issue I had before. You can see the radio frequency and sunspot number lead by a clear margin, reflecting overall solar activity. Next comes the solar wind speed and other geomagnetic features.

**Slide 17**

So I showed that the Transformer was able to push the 24-hour flare prediction well beyond static baselines and am working on an inference pipeline that can work in real-time. In terms of future work, though the project was a proof of concept it is clear that precision is still very low at 11%. With the current threshold I have it set at, the false-alarm rate is extremely high, however that’s better than it being flipped and the no alarm rate is high. To improve on this, I can include solar imagery to let the model ‘see’ evolving active-region complexity instead of inferring it indirectly from radio flux. A small CNN can turn each image into a vector that’s then appended to the 5-minute sequencing before the transformer. In the future I’d also want to allow the model to predict storm severity alongside flares, which some experiments in the literature are doing.

**Q1: What exactly was your validation set?**

My validation split consisted of all 48‑hour windows drawn from **January 1, 2014 through December 31, 2016**. In practice that meant after building the full 5‑minute grid (2000 – 2025) and labeling each window’s “flare\_next\_24h,” I chunked the data chronologically into:

* **Train:** up to 2013‑12‑31
* **Val:** 2014‑01‑01 – 2016‑12‑31
* **Test:** 2017‑01‑01 – 2018‑12‑31

The **2014–2016** set (≈ 26 k sequences at 5‑min cadence) was used for early stopping (monitoring PR‑AUC), hyperparameter tuning, and threshold selection—but it was never touched during final test evaluation.

**Q2: Did you run into any hardware issues when training?**

No major hardware bottlenecks. I ran both the Bi‑LSTM and Transformer on an Azure GPU VM using my free credits. Training each epoch (~800 s per epoch for the LSTM; ~1 650 s for the Transformer) fit comfortably within the VM’s memory and compute limits, and I used early stopping to avoid wasted time on later epochs.

Q3: On determining threshold   
Since flares are so rare, the model outputs are usually tiny. I found that setting the threshold to 0.01 gave the best F1 score — it helps catch subtle but meaningful signals without letting false positives overwhelm the system.”