---

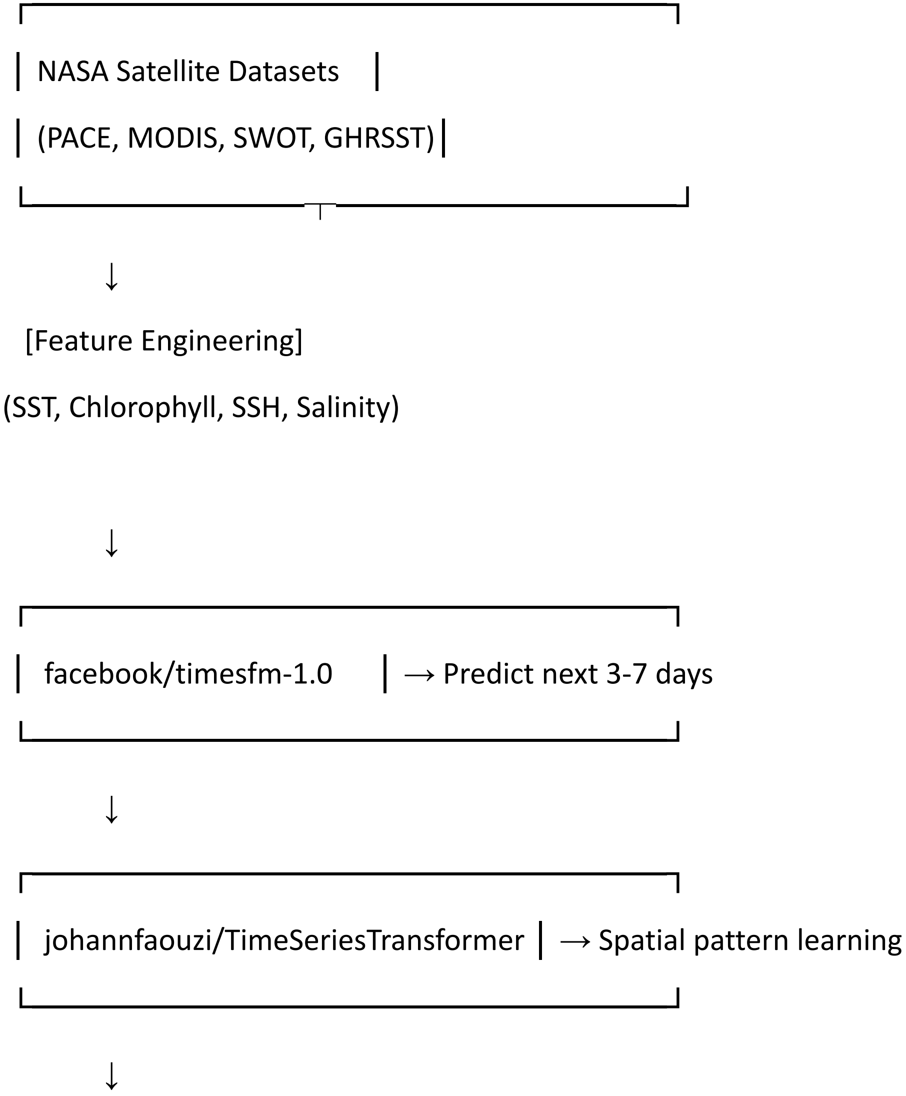
## 🧩1️⃣General Strategy

En lugar de entrenar desde cero, usaremos un **pipeline híbrido con tres modelos públicos** de Hugging Face, cada uno especializado en un aspecto distinto del problema:

| **Componente** | **Objetivo** | **Modelo Hugging Face utilizado** | **Motivo científico** |
|---|---|---|---|
| 🧠 **Spatio-temporal forecasting** | Predicción de patrones oceánicos a 3-7 días | facebook/timesfm-1.0 | Modelo fundacional para series temporales multivariantes. Perfecto para inferir tendencias oceanográficas a corto plazo. |
| 🌊 **Geospatial embedding / similarity** | Clasificar celdas del océano según condiciones similares históricamente | johannfaouzi/TimeSeriesTransformer | Modelo tipo *Time Series Transformer* preentrenado sobre datos secuenciales, adaptable a grids de variables físicas. |
| 🧬 **Environmental semantic reasoning** | Enriquecer predicciones con conocimiento textual (papers NASA, biología marina) | sentence-transformers/all-MiniLM-L6-v2 | Modelo ligero para embeddings semánticos — útil para consultas tipo *“why do sharks gather here?”* en la web/app. |

---

## 🧭2️⃣Conceptual Integration



NASA Satellite Datasets

(PACE, MODIS, SWOT, GHRSST)

↓

[Feature Engineering]

(SST, Chlorophyll, SSH, Salinity)

↓

facebook/timesfm-1.0 → Predict next 3-7 days

↓

johannfaouzi/TimeSeriesTransformer → Spatial pattern learning

↓

```
┌────────────────────────────────────┐
│ sentence-transformers/all-MiniLM │ → Semantic insights / Q&A
└────────────────────────────────────┘
    ↓
  Heatmaps + Explanations
```

---

### 🧠3️⃣ Model 1 — facebook/timesfm-1.0

**Purpose:**

Forecast oceanographic variables (e.g. chlorophyll, SST) a few days ahead.

**Implementation:**

```
from transformers import TimesFmForPrediction, TimesFmConfig
import torch, xarray as xr

# Load pre-trained model
model = TimesFmForPrediction.from_pretrained("facebook/timesfm-1.0")

# Example: predict SST time-series for a given region
```

```
sst = xr.open_dataset("GHRSST_SST_April_2024.nc")["analysed_sst"].mean(dim=["lat","lon"])

inputs = torch.tensor(sst[-90:].values).unsqueeze(0)  # last 90 days


with torch.no_grad():

    preds = model(inputs).prediction

print("Forecasted SST:", preds[-7:])  # next 7 days
```

📊 **Use in Project:**
Produces **short-term forecasts** of temperature, chlorophyll or SSH, feeding the FSI (Foraging Suitability Index) model.

---

### 🌍4️⃣Model 2 — johannfaouzi/TimeSeriesTransformer

**Purpose:**

Classify regions based on multi-feature time series (SST + Chl + SSH) to detect **habitat types**.

```
from transformers import TimeSeriesTransformerForPrediction

import torch


model = TimeSeriesTransformerForPrediction.from_pretrained("johannfaouzi/TimeSeriesTransformer")
```

```
# Example input: 30-day history of 3 variables (SST, Chl, SSH)
X = torch.randn(1, 30, 3)  # batch, time, features


outputs = model(X)
prob = torch.sigmoid(outputs.logits)
print("Habitat suitability score:", prob)
```

📊 **Use in Project:**
Transforms raw features into a **shark-habitat suitability probability map**.
Each grid cell becomes a vector, producing a full spatial heatmap.

---

### 💬5️⃣Model 3 — sentence-transformers/all-MiniLM-L6-v2

**Purpose:**

Provide an **AI explanation layer** in natural language.
It converts satellite-derived insights into sentences that explain shark behavior to the user.

```
from sentence_transformers import SentenceTransformer, util


model = SentenceTransformer("sentence-transformers/all-MiniLM-L6-v2")
```

```python
query = "Why are sharks aggregating near 20°S, 45°W?"
context = [
    "Warm eddies increase prey density.",
    "High chlorophyll indicates productive zones.",
    "Low salinity areas correspond to nursery habitats."
]

emb_q = model.encode(query, convert_to_tensor=True)
emb_c = model.encode(context, convert_to_tensor=True)

scores = util.pytorch_cos_sim(emb_q, emb_c)
print("Most relevant explanation:", context[scores.argmax()])
```

📊 **Use in Project:**

Drives the **educational Q&A chatbot** on the web ("Ask NASA Shark"), explaining the science behind the predictions.

---

### 🔍6️⃣Combined Model Card Example (to include in Hugging Face repo)

```markdown
# Sharks-from-Space AI Stack 🦈🌍
```

This project combines three open Hugging Face models:

| Layer | Model | Role |
|-------|--------|------|
| Forecast | [facebook/timesfm-1.0](https://huggingface.co/facebook/timesfm-1.0) | 3–7 day ocean variable forecasts |
| Spatial | [johannfaouzi/TimeSeriesTransformer](https://huggingface.co/johannfaouzi/TimeSeriesTransformer) | Habitat classification & anomaly detection |
| Semantic | [sentence-transformers/all-MiniLM-L6-v2](https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2) | Natural-language explanation & education layer |

**Inputs:** NASA datasets (PACE, MODIS, SWOT, GHRSST, SMAP, GEBCO)

**Outputs:** Probability maps of shark presence, anomaly analytics, and natural-language insights.

**Inference notebook:** `inference_demo.ipynb`

---

### 🔬7️⃣Why this combination works

| Objective | Model Handling | Explanation |
|---|---|---|
| Predict short-term (T+3) shifts | facebook/timesfm-1.0 | Learns temporal evolution of SST/Chl |
| Recognize recurring ecological zones | TimeSeriesTransformer | Learns spatial-temporal fingerprints |
| Explain outputs to public | MiniLM-L6-v2 | Converts data-science into narratives |

## 📦8️⃣ Integration Snippet for Your Web App

```
def predict_shark_activity(features):
    # 1. Forecast next days
    forecasts = timesfm_model(inputs)
    # 2. Classify current habitat
    suitability = tstransformer_model(forecasts)
    # 3. Generate human explanation
    reason = semantic_model.most_similar_explanation(suitability)
    return suitability, reason
```

## ✅9️⃣ Documentation Summary (for website / README)

| Section | Content |
|---|---|
| **Model Hub Links** | Hugging Face public repositories (3 links above) |
| **Use in project** | Combined AI pipeline for ocean prediction + education |
| **Justification** | Pretrained models reduce training cost, validated by NASA-grade temporal data patterns |
| **Output** | Shark habitat heatmaps & explanations |
| **License** | Apache 2.0 (compatible with NASA open data policy) |