PRINCETON UNIVERSITY

Convolutional Transformers for Inertial Navigation

Raymond Liu — rl27@princeton.edu
Adviser: Prof. Olga Russakovsky

CENTER FOR STATISTICS AND MACHINE LEARNING

Motivation and Goal

An **Inertial Measurement Unit (IMU)** uses an accelerometer and a gyroscope to measure a body's linear acceleration and angular velocity in 3 dimensions.

IMUs are cheap, energy-efficient, ubiquitous, and have many applications, such as navigation for robots and pedestrians, as well as augmented reality systems.

Inertial navigation is the task of estimating a body's trajectory using IMU sensor measurements. This task is inherently difficult, as small amounts of noise in these measurements accumulate over time to large errors in position estimates.

Several different methods have been developed to deal with this. Deep learning methods have been particularly effective and achieve state-of-the-art results.

The goal of this work is to introduce transformer-based models that outperform the existing best models. We implement and evaluate transformer-based models that outperform the existing best methods.

Related Work

PDR[2] (2015): Detect footsteps, estimate step length, and update location at every step. PDR fails with complex motion cases, such as walking sideways.

RIDI[3] (2017): First machine learning method. Uses statistical learning to classify device placement, then regress velocities to obtain positions. RIDI only works for four specific device placements and is thus not robust to all use cases.

IONet[4] (2018): Uses LSTM, an RNN architecture, to regress velocity and orientation. Outperforms previous methods but suffers from accumulated orientation drift on noisy data.

RoNIN Model and Dataset

RoNIN[1] (2019) introduced a 1D variant of ResNet, a CNN architecture, to regress velocities and obtain positions. RoNIN ResNet outperforms all previous methods – we use RoNIN ResNet as a baseline model to compare against other models.

RoNIN also introduced a comprehensive, diverse dataset of IMU data and ground truth positions, which we use to train and test different models.

Approach

Transformers[5] are a highly effective neural network architecture that achieve better performance than RNNs and CNNs on a variety of tasks, including machine translation, image classification, and object detection.

We first implement a model that uses a transformer encoder to use as a baseline, then introduce convolutions to improve baseline transformer.

Transformer Architecture

The baseline transformer model consists of 6 transformer encoders followed by an MLP layer. Each encoder contains a multi-head self-attention unit followed by a feedforward unit. Self-attention is a powerful method of encoding a representation of an input sequence. Given an input *X*, self-attention *S* is calculated simultaneously for every head *i*.

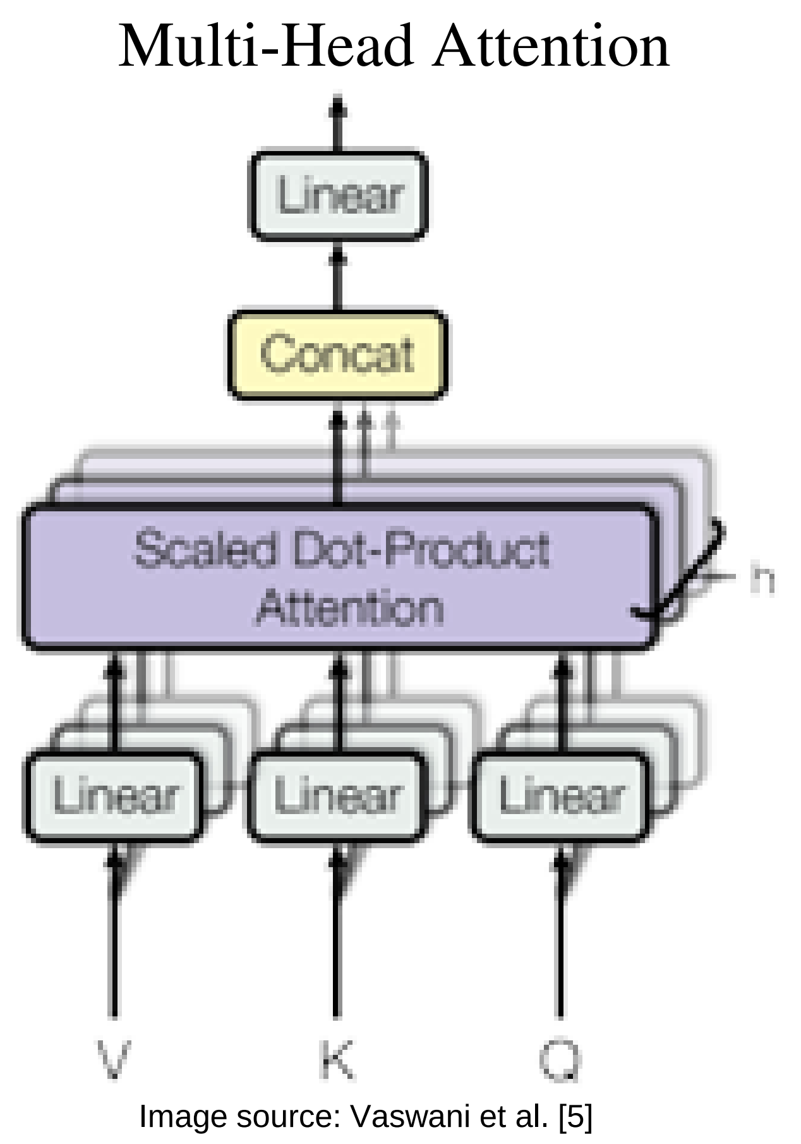
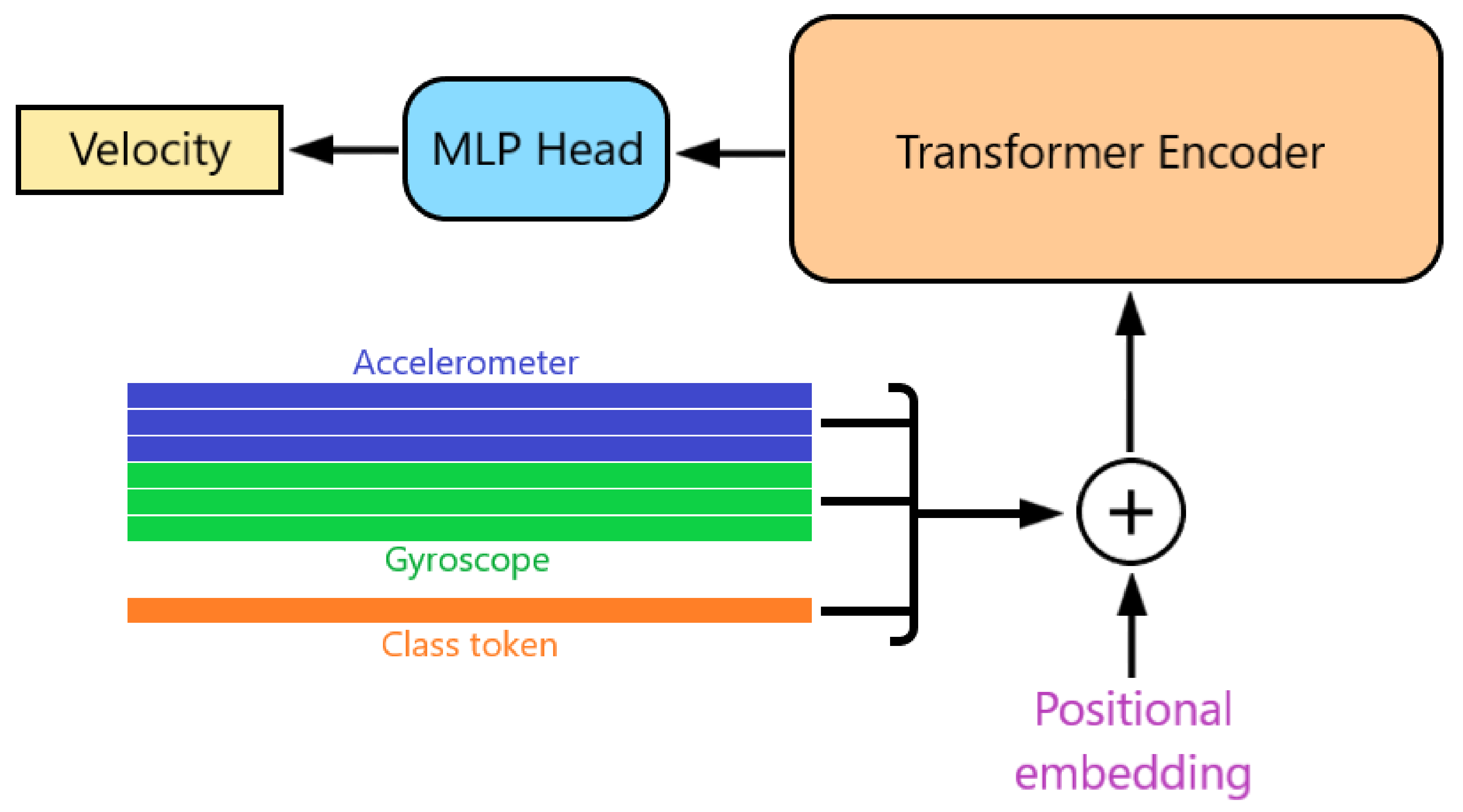


Image source: Vaswani et al. [5]

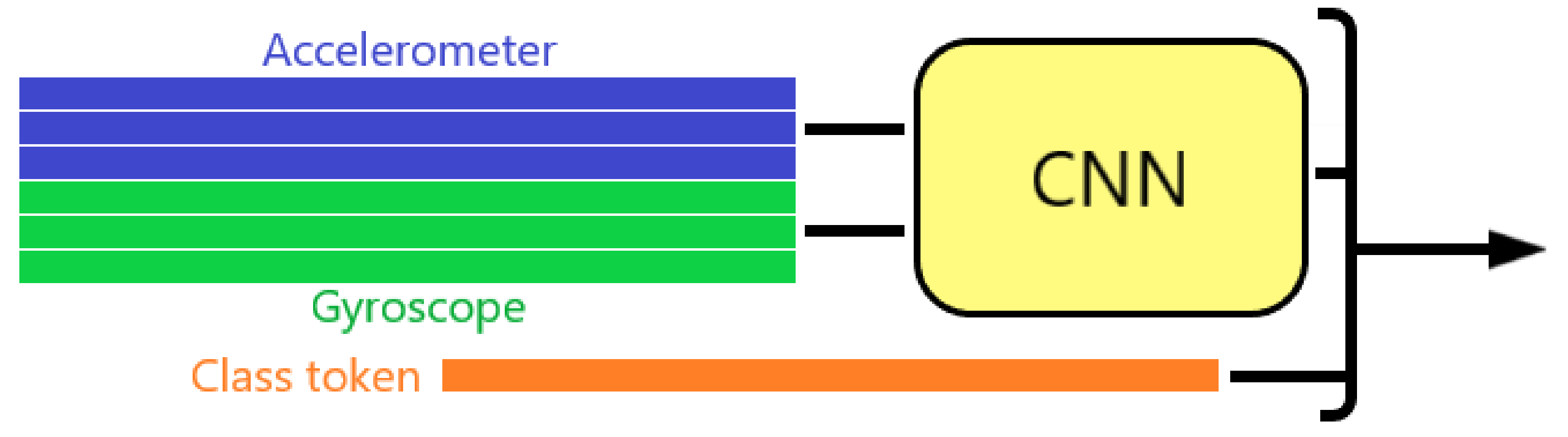
$$X \in \mathbb{R}^{6\times 200}$$
$$\downarrow$$
$$q_i = XW_i^q,\quad k_i = XW_i^k,\quad v_i = XW_i^v$$
$$\downarrow$$
$$S_i = \mathrm{softmax}\left(\frac{q_i k_i^\top}{\sqrt{d_h}}\right) v_i$$

The input consists of the past 200 frames of 6-dimensional IMU data, and the output is a 2D vector representing the velocity at the current frame. Regressed velocities are summed to reconstruct relative positions.



Convolutional Transformer

To improve on the baseline transformer, we introduce convolutional layers to help extract local information for the transformer. The most effective convolutional transformer model applies multiple convolutional layers to the input sequentially before the transformer.



Results

We implemented and evaluated two baseline models and three convolutional transformer models, as well as several variations for each model. The main convolutional transformer models are:

- CNN stacked with input: Conv layers are applied separately to the accelerometer and gyroscope data, and the output is used alongside the original input as an input to the transformer.
- CNN sequential to transformer: As previously described, this model applies a CNN to the input sequentially before the transformer.
- CNN parallel to transformer: Applies the same CNN, but in parallel to the transformer, then concatenates the results from the CNN and the transformer afterwards.

Evaluation is done using two metrics:

- Absolute Trajectory Error (ATE): the root mean squared error between the ground truth trajectory and the estimated trajectory
- Relative Trajectory Error (RTE): the average root mean squared error over fixed time intervals of 1 minute

Models are evaluated on the RoNIN testing set. Results for the main models are shown below.

Model	ATE	RTE
Baseline ResNet	5.61	4.49
Baseline Transformer	5.12	4.40
Conv Stacked with Input	5.03	4.31
CNN Sequential to Transformer	4.99	4.24
CNN Parallel to Transformer	5.26	4.49

Conclusions

Transformers work well with IMU data and perform better than previous approaches for the task of inertial navigation, including CNNs such as ResNet and RNNs such as LSTM.

Incorporating convolutions into transformers can further improve performance, although they can also possibly degrade performance.

Our findings present a new pathway of research into applications of transformers for inertial navigation.

References

[1] Herath et al. "RoNIN: Robust neural inertial navigation in the wild: Benchmark, evaluations, new methods." IEEE ICRA, 2020.
[2] Tian et al. "An enhanced pedestrian dead reckoning approach for pedestrian tracking using smartphones." IEEE ISSNIP, 2015.
[3] Yan et al. "RIDI: robust IMU double integration." ECCV, 2018.
[4] Chen et al. "IONet: Learning to cure the curse of drift in inertial odometry." AAAI, 2018.
[5] Vaswani et al., "Attention is all you need." NIPS, 2017.