Recovering Predictability from the Implied Volatility Surface: A Transformer Approach*

November 17, 2025

Abstract

Placeholder

Keywords: key1, key2, key3

JEL Codes: key1, key2, key3

*abc

1 Introduction

The information contained in options market has been widely studied since the birth of options market. There exists a series of well-developed theories that uncover the risk-neutral, as well as physical or real probabilities of the underlying asset returns from their option prices (Breeden and Litzenberger, 1978; Cox and Ross, 1976a,b; Ross, 2015). However, the practical implementation of extracting the information from option prices is often challenging, due to the discrete nature of option strike prices and maturities, limited liquidity of option market, as well as the model misspecification issues.

Option-implied volatilty surface (IVS), as the most prominent representation of option prices, has been shown to have significant predictive power for future stock returns in the empirical options literature. For example, Yan (2011) showed that the spread between the implied volatilities of at-the-money (ATM) put and call with 30 days to expiration can predict the stock returns over the next month. Xing et al. (2010) also found that the volatility skew, as measured by the difference between implied volatilities of out-of-the-money (OTM) put and at-the-money (ATM) call with the same maturity, has the ability to forecast stock returns. Other studies have also documented a variety of option-based predictors, mostly derived from the implied volatility surface. [1]

However, most of the existing option-based predictors are just transformations of a small subset of the implied volatility surface, which might not fully capture the rich information embedded in the entire surface. Moreover, the linear transformations used in most predictors may not be sufficient to extract the complex, non-linear patterns from the high-dimensional volatility surface data. Therefore, it is natural to ask whether we can develop more advanced methods to recover the information contained in the entire implied volatility surface, and whether such information can provide additional predictive power for future stock returns.

Recent years have witnessed the rapid development of machine learning and deep learning

[1] See Bali and Hovakimian (2009); Martin (2017); Martin and Wagner (2019); An et al. (2014); Baltussen et al. (2018) etc.

techniques, and their applications in the research of asset pricing fields (Gu et al., 2020; Bianchi et al., 2021; Jiang et al., 2023). These advanced methods have shown great potential in extracting complex, non-linear patterns from high-dimensional, and unstructured data, which are often difficult to capture using traditional econometric models. Motivated by these advancements, we propose to leverage the well known deep learning architecture, namely the Transformer model, to extract the information from the option-implied volatility surface and predict future stock returns.

Since the work of Vaswani et al. (2017), the Transformer model has revolutionized the field of natural language processing (NLP) and has been widely adopted in various applications, such as machine translation, text summarization, and sentiment analysis. The core component of the Transformer architecture is the self-attention mechanism, which allows the model to weigh the importance of different words in a sentence when making predictions. This mechanism enables the model to capture long-range dependencies and contextual information effectively, leading to significant improvements in performance compared to previous models, such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs). Building on the success of the Transformer, large language models (LLMs), such as GPT series (Brown et al., 2020) and BERT (Devlin et al., 2019), have further pushed the boundaries of what is possible with deep learning in NLP, achieving state-of-the-art results on a wide range of tasks.

In this paper, we leverage the vanilla Transformer model and its variant in computer vision, namely the Vision Transformer (ViT) (Dosovitskiy et al., 2020), to construct non-linear local feature representations from the option-implied volatility surface data. In the first approach, we treat the volatility surface as a time series of flattened vectors over a certain "lookback" window, and use the Transformer encoder to capture the temporal dynamics of the volatility surface. In the second approach, we treat the volatility surface as a single-channel image, and use the ViT model to extract the spatial structure of the volatility surface.

The implied volatility surface, expressed as a two-dimensional matrix across different maturities and moneyness levels, can be naturally represented as a single-channel (e.g., grayscale) image, where the height corresponds to the number of maturities and the width corresponds to the number of moneyness levels. This representation allows us to leverage the powerful computer vision and image processing techniques developed in the deep learning literature. Kelly et al. (2023)

2 Data

2.1 Option-Implied Volatility Surface (IVS)

We obtain the equity and index option data from the IvyDB OptionMetrics database, which provides the files (vsurfdYYYY) that contain the interpolated Black-Scholes option-implied volatilities, starting from January 1996. For each security on each day, a volatility surface with standard maturities of τ days to expiration and moneyness levels measured by option δ is provided. [2]

We select out-of-the-money (OTM) and near at-the-money (ATM) put options with delta levels in [-0.5, -0.1], as well as OTM and near ATM call options with delta levels in [0.1, 0.5], since the deep-in-the-money call options are often illiquid, following Martin (2017), among others. The options are rearranged by their moneyness (implied strike), so the delta levels start from -0.1 to -0.5, then from 0.5 to 0.1 [3]. We also drop the options with maturities equal to 10 days, since the large fraction of missing values. The final volatility surface data contains 10 maturities and 18 delta levels, and a example of the volatility surface is shown in Figure 2. The implied volatility surface of each security i at time t can be represented as

[2] OptionMetrics use a methodology based on a kernel smoothing algorithm to interpolate the volatility surface. The maturities are 10, 30, 60, 91, 122, 152, 182, 365, 547, 730 days to expiration, and the option delta levels are from 0.1 to 0.9 with a step of 0.05, positive for call options and negative for put options.

[3] The implied strike order between put option with delta -0.5 and call option with delta 0.5 is uncertain, but their spread is proved to have predictive power for future returns (Yan, 2011), so we keep both of them.

a collection of volatility values at different maturities and delta levels:

$$
\begin{aligned}
IV_{i,t} &= \{IV_{i,t}(\tau,\delta)\}_{\tau\in T,\delta\in\Delta} \\
T &= \{30, 60, 91, 122, 152, 182, 365, 547, 730\} \\
\Delta &= \{-0.5, -0.45, ..., -0.1, 0.1, 0.15, ..., 0.5\}
\end{aligned}
\tag{1}
$$

Where $IV_{i,t}(\tau,\delta)$ represents the implied volatility level at maturity τ and delta level δ for security i at time t, which serves as the main input feature in our Transformer models.

We obtain the daily return data from CRSP for all firms listed on NYSE, AMEX, and NASDAQ, as well as the S&P 500 index. For each security i at time t, we calculate the cumulative return on security i from day t to day $t+H$ as:

$$
r_{i,t}^{H} = \prod_{j=1}^{H}(1 + r_{i,t+j}) - 1 \tag{2}
$$

We link the volatility surface data from OptionMetrics with the return data from CRSP using a linking table provided by WRDS. The final dataset spans from January 1996 to August 2023, and Figure 1 shows the coverage of the stocks with available volatility surface and return data over time. The number of stocks increases significantly over time, from below 1000 to over 5000 in recent years.

2.2 Option-Based and Other Characteristics

Motivated by the extensive literature from cross-section studies of stock returns, we also collect a set of option-based predictors that have been shown to have predictive power for future stock returns. Neuhierl et al. (2022) and Muravyev et al. (2025) summarized a comprehensive list of such option characteristics and compared their performance. We follow their results and select the strongest ones. Other common stock characteristics are also collected. A detailed description of these characteristics is provided in Appendix B.

3 The Transformer Model

In this section, we briefly introduce the vanilla Transformer model and Vision Transformer (ViT) model architectures and our experimental designs.

3.1 Brief Introduction of the Transformer Encoder and Vision Transformer

First introduced by Vaswani et al. (2017), "Attention Is All You Need," the Transformer model has become a foundational architecture in deep learning, particularly for natural language processing (NLP). Unlike its predecessors, such as Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, the Transformer does not rely on sequential data processing. Instead, it processes the entire input sequence at once, using a sophisticated mechanism known as self-attention to weigh the importance of different words in the sequence.

The Transformer architecture is depicted in Figure 4. The core of the model is its encoder-decoder structure. The encoder's role is to map an input sequence of symbol representations $(x_1, ..., x_n)$ to a sequence of continuous representations $\mathbf{z} = (z_1, ..., z_n)$. The decoder then takes $\mathbf{z}$ and generates an output sequence $(y_1, ..., y_m)$, one symbol at a time. In our study, we want to leverage the Transformer architecture for predicting stock returns based on the option-implied volatility surface, instead of the time series self-prediction tasks. Therefore, we focus on the Transformer encoder part of the architecture, that is, the left half of Figure 4, which is responsible for encoding the input sequence into a dense vector representation. The so-called "Encoder-Only" architecture, has been widely used in various applications, such as BERT (Devlin et al., 2019) for language understanding and other domains beyond NLP.

Building on the success of the Transformer in NLP, the Vision Transformer (ViT) was proposed by Dosovitskiy et al. (2020) to apply the same architecture to computer vision tasks, offering a compelling alternative to the widely used Convolutional Neural Networks (CNNs). The fundamental idea behind ViT is to treat an image as a sequence of patches, analogous to how a sentence is treated as a sequence of words. By converting images into a sequential format, ViT demonstrates that the reliance on convolutions is not a necessity for vision tasks and that a general-purpose attention-based architecture can achieve state-of-the-art (SOTA) performance, especially when pre-trained on large datasets.

In the computer vision (CV) field, a image is typically represented as a three-dimensional matrix with (C, H, W) dimensions, where C is the number of channels (e.g., $C = 3$ for RGB images, and $C = 1$ for grayscale images), H is the height (number of pixels in vertical direction), and W is the width (number of pixels in horizontal direction). Each pixel in the image has a value ranging from 0 to 255, representing the intensity of the color channel(s) at that pixel. In the ViT model, the input image is first divided into fixed-size patches (e.g., 16×16 pixels), and each patch is then flattened into a vector. These patch vectors are linearly embedded into a higher-dimensional space, and positional encodings are added to retain spatial information. The resulting sequence of embedded patches is then fed into a standard Transformer encoder, similar to the original Transformer architecture used in NLP. The ViT model can be visualized as shown in Figure 5.

We consider two different model structures, mainly differing in the way of representing the implied volatility surface data as input features, that is, the ”embedding” stage of the Transformer model. The next section describes the two different embedding methods in detail, and the details of the model architecture are provided in Appendix A.

3.2 Embedding of the Implied Volatility Surface

Time Series Representation

The first method treats the volatility surface as a time series of flattened vectors over the past month. At each time t, the volatility surface of security i can be represented as a two-dimensional matrix with dimensions (N_τ, N_δ), where N_τ is the number of maturities and N_δ

is the number of moneyness levels. We then flatten the matrix into a vector of size $N_\tau \times N_\delta$. To capture the time series dynamics of the volatility surface, we collect the volatility surfaces over the past B days (e.g., $B = 21$ trading days for one month), and stack them together to form a sequence of vectors $(x_1, x_2, ..., x_B)$, where each vector $x_j \in \mathbb{R}^{N_\tau \times N_\delta}$ represents a flattened volatility surface at time step j in the lookback window.

We use a linear embedding layer to project the input vectors into the model dimension d_{model}, followed by adding a positional encoding to retain the temporal information[4]. The embedded sequence is then fed into the Transformer encoder, and we select the last temporal output in the last layer as the dense vector representation of the volatility surface information[5]. The model can be simply expressed as:

$$\mathbf{v}_{i,t} = \text{Transformer}([IV_{i,t-B+1}; IV_{i,t-B+2}; ...; IV_{i,t}]|\theta) \tag{3}$$

Where $IV_{i,t}$ represents the implied volatility surface of security i at time t, and $\mathbf{v}_{i,t} \in \mathbb{R}^{d_{model}}$ is the output dense vector representation of the information contained in the volatility surfaces, and θ is the set of parameters needed to be learned through training. $\text{Transformer}(\cdot|\theta)$ represents the non-linear mapping from the input sequence to the output vector through the linear embedding, positional encoding and the Transformer encoder, while not including the final classification head. This approach treats each volatility surface as a token and construct the sequence based on the time series order. The benefit of this embedding methodology is its ability to capture the temporal dynamics of the volatility surface over a certain period, while the weakness is that it ignores the spatial structure of the volatility surface itself. We refer to this model as the **TF** (Temporal Transformer) model in the following sections.

Besides the implied volatility surface of individual stocks, we also consider to incorporate the implied volatility surface of the S&P 500 index options as an additional input. The

[4]We use the sinusoid function as in Vaswani et al. (2017), see Appendix A for details

[5]Alternatively, we can also use the mean pooling of all temporal outputs in the last layer $\frac{1}{21}\sum_{j=0}^{20} \mathbf{z}_L^j$ as the vector representation. Since the sequence here is a time series, the last temporal output is more intuitive and might contain more recent information.

index options contain essential information of market-wide risk and risk premia, in time series as well as in cross-section, providing incremental predictive power for the stock returns (Andersen et al., 2015; Ang et al., 2006). We simply concatenate the flattened index volatility surface with the flattened stock volatility surface along the feature dimension, resulting in an input vector of size $(B, 2 \times N_\tau \times N_\delta)$, and the model can be expressed as:

$$\mathbf{v}_{i,t} = \text{Transformer}([(IV_{i,t-B+1}, IV_{spx,t-B+1}); ...; (IV_{i,t}, IV_{spx,t})]|\theta) \tag{4}$$

Where $IV_{spx,t}$ represents the implied volatility surface of the S&P 500 index at time t.

Image Representation

An option-implied volatility surface can be represented as a three-dimensional plot that displays the implied volatility of a security options across different strike prices and expirations (Figure 2). Therefore, the volatility surface is analogous to a single-channel grayscale image: The height H of the matrix corresponds to the number of maturities N_τ, and the width W corresponds to the number of moneyness levels N_δ (Figure 3).

We implement a standard Vision Transformer (ViT) model to extract the information from the volatility surface. Each volatility surface $IV_{i,t}$ is reshape as a three-dimensional matrix with dimensions $(1, N_\tau, N_\delta)$, where the first dimension represents the single channel of the grayscale image. The ViT model splits the input image into fixed-size patches. In this study, we simply set the patch size to be $(1, 1)$, meaning that each patch contains a single volatility value in the surface. Each patch (volatility value) is linearly embedded into a higher-dimensional space (d_{model}), and learnable positional encoding is added to retain the spatial information, as well as a learnable classification token (CLS token) is prepended to the sequence of embedded patches (see more details in Appendix A). The resulting sequence of embedded patches is then fed into a standard Transformer encoder. Finally, the output corresponding to the CLS token in the last layer is selected as the dense vector representation of the volatility surface information. The model can be simply expressed as:

$$\mathbf{v}_{i,t} = \mathbf{ViT}(IV_{i,t}|\theta) \tag{5}$$

Where the $ViT(\cdot|\theta)$ denote the non-linear mapping from the input image (volatility surface) to a dense vector representation of the image, except for the final classification head. We refer to this model as the **ViT** (Vision Transformer) model in the following sections.

We also consider to incorporate the index volatility surface as the additional input, and it's natural to treat the index volatility surface as another channel of the input image. However, unlike standard RGB images, where the three channels (Red, Green, Blue) contain correlated and complementary information about the same object, the two implied volatility surfaces in our case contain semantically distinct and independent information. This poses a challenge for the standard ViT model, which is designed to process images with correlated channels. Inspired by Bao et al. (2024), we use the so-called ChannelViT model for the embedding of the multiple volatility surfaces. Specifically, the ChannelViT model, shown in Figure 6, processes each input channel (volatility surface) independently in the embedding stage. Each channel (volatility surface) is split into patches, linearly embedded, and employs a set of learnable channel embeddings to encode the channel-specific information besides the positional embeddings. The resulting sequence of vectors from all channels is then concatenated and fed into the Transformer encoder. Since the two volatility surfaces have the exactly the same structure (same maturities and moneyness levels), they share a common positional encoding matrix. The model can be expressed as:

$$\mathbf{v}_{i,t} = \mathbf{ViT}([IV_{i,t}, IV_{spx,t}]|\theta) \tag{6}$$

Where $[IV_{i,t}, IV_{spx,t}]$ represents the two-channel volatility surface image, with the first channel being the stock volatility surface and the second channel being the index volatility surface at the same time t.

Temporal-Spatial Hybrid Representation

A natural extension of the above two embedding methods is to combine both the temporal dynamics and spatial structure of the volatility surface, which regard the volatility surface data as a sequence of images over time, similar to a video data. For example, this can be achieved by stacking the volatility surfaces over the past B days as multiple channels of a single input image, and implementing a ChannelViT model to extract the information. However, this approach would lead to a very long sequence of inputs, resulting in high computational costs and memory requirements.

Alternatively, we can utilize more advanced architectures that combine the strengths of both temporal and spatial modeling, such as TimeSformer (Bertasius et al., 2021) and Video Vision Transformer (ViViT) (Arnab et al., 2021), which are specifically designed for video data. These models employ specialized attention mechanisms to effectively capture both temporal and spatial dependencies in the data. However, these higher-complexity models require significantly more computational resources and larger datasets for training, which may not be feasible in our current setting. Therefore, we leave the exploration of these hybrid models for future research.

3.3 Discretization of Returns

Utilizing the dense vector representation $\mathbf{v}_{i,t}$ obtained from the ViT model, a classification head is attached to perform classification tasks. In our study, such a classification head is a single linear layer followed by a softmax function, as shown in Figure 4 and 5 [6], which maps

[6] In the original ViT paper, Dosovitskiy et al. (2020) used a multi-layer perceptron (MLP) with one hidden layer at pre-training time and used a single linear layer at fine-tuning time. A recent update, Beyer et al. (2022) from some of the same authors of the original paper suggests that a simple linear layer at the end is not significantly worse than an MLP. We use a single linear layer for simplicity reason.

the vector representation $\mathbf{v}_{i,t}$ to a probability distribution over the classes:

$$\mathbf{p}_{i,t} = \text{softmax}(W_c \cdot \mathbf{v}_{i,t} + b_c) \tag{7}$$

$$\text{softmax}(\mathbf{x})_j = \frac{e^{x_j}}{\sum_{k=1}^{K} e^{x_k}}, \quad j = 1, ..., K \tag{8}$$

where $W_c \in \mathbb{R}^{K \times d_{model}}$ and $b_c \in \mathbb{R}^K$ are the weights and bias of the linear layer in the classification head, and K is the number of classes. The output $\mathbf{p}_{i,t} \in \mathbb{R}^K$ is a probability vector, where each element $\mathbf{p}_{i,t}(k)$ represents the predicted probability of stock i's return falling into class k at time t.

The simplest classification is the binary classification, where the number of classes $K = 2$. For example, we can classify the stock return into "up" and "down" classes based on whether the future return is positive or negative, as in Jiang et al. (2023). Alternatively, the stock return can be classified into "safe" and "crash" classes based on whether the future return is above a certain negative value (e.g., -10%). In general, we denote $y_{i,t}$ as the binary variable indicating whether stock i's return falls into the two classes defined by a specific threshold q at time t:

$$y_{i,t} = \mathbf{I}(r_{i,t}^H > q) = \begin{cases} 1, & \text{if } r_{i,t}^H > q \\ 0, & \text{if } r_{i,t}^H \leq q \end{cases} \tag{9}$$

$$\hat{y}_{i,t} = \mathbf{p}_{i,t}(1) \tag{10}$$

where $r_{i,t}^H$ is the cumulative return of stock i from day t to day $t + H$, and the output $\hat{y}_{i,t} = \mathbf{p}_{i,t}(1)$ represents the predicted probability of stock i's return being in class 1 at time t. The threshold q can be set to 0 for the "up/down" classification. [7]

For multi-class classification ($K > 2$), the stock returns can be discretized into multiple classes based on quantiles or specific thresholds. Assume $\{q_0, q_1, ..., q_K\}$ are the thresholds

[7] In the special case of binary classification ($K = 2$), the softmax function reduces to the logistic (sigmoid) function.

that define the K classes, where $q_0 = \min(r)$ and $q_K = \max(r)$. For example, we can classify the stock returns into quintiles ($K = 5$) based on the 20th, 40th, 60th, and 80th percentiles of the return distribution.

In general, we denote $y_{i,t}$ as the categorical variable indicating which class stock i's return falls into at time t:

$$y_{i,t} = k, \quad \text{if } r_{i,t}^H \in (q_{k-1}, q_k], \quad k = 1, 2, ..., K \tag{11}$$

$$\hat{y}_{i,t}^k = \mathbf{p}_{i,t}(k) = \hat{P}\left(r_{i,t}^H \in (q_{k-1}, q_k]\right) \tag{12}$$

3.4 Training Process

We use standard cross-entropy loss function for the classification task. For binary classification, where $y \in \{0, 1\}$, the loss function can be expressed as:

$$\mathcal{L}(y, \hat{y}) = -\left[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})\right] \tag{13}$$

For multi-class classification, where $y \in \{1, 2, ..., K\}$, the loss function can be expressed as:

$$\mathcal{L}(y, \hat{y}) = -\sum_{k=1}^{K} \mathbf{I}(y = k) \log(\hat{y}^k) \tag{14}$$

where $\hat{y}^k$ is the predicted probability of class k.

In the case of imbalanced classes, we can use a weighted cross-entropy loss function to give more importance to the minority classes. The weighted loss function can be expressed as:

$$\mathcal{L}(y, \hat{y}) = -\sum_{k=1}^{K} w_k \cdot \mathbf{I}(y = k) \log(\hat{y}^k) \tag{15}$$

$$w_k = \frac{N}{K \cdot N_k} \tag{16}$$

where w_k is the weight for class k, N is the total number of samples, and N_k is the number of samples in class k. This weighting scheme helps to balance the influence of each class during training, especially when some classes are underrepresented.

We adopt similar regularization techniques as in Gu et al. (2020), to prevent overfitting and improve the comuputational efficiency. We use a Adaptive Moment Estimation With Weight Decay (AdamW) optimizer (Loshchilov and Hutter, 2019), which is standard in the transformer training and a refined version of the Adam optimizer (Kingma and Ba, 2017). AdamW decouples the weight decay (L2 regularization) from the gradient update, which has been shown to lead to better generalization performance. In this study, we set the learning rate to 1×10^{-5}, the weight decay to 1×10^{-2} and the batch size to 512. We use a dropout rate of 0.2 to prevent overfitting. Dropout is a regularization technique that randomly sets a fraction of the input units to zero during training, which helps prevent overfitting by reducing the model's reliance on specific features.

We apply multiple methods for parameter initialization: for weights in the patch embedding layers and classfication head, we use the Xavier initialization (Glorot and Bengio, 2010), which is the best practice for deep neural networks such as CNNs; for parameters in the CLS token and positional embeddings, as well as the weights in the Transformer encoder, we use a truncated normal distribution with a standard deviation of 0.02, following the BERT model Devlin et al. (2019); for layer normalization parameters, we initialize the weights to 1; the biases in all layers are initialized to 0. We use a standard "warm-up" and "cosine decay" learning rate scheduler, which gradually increases the learning rate from zero to the initial value over a specified number of warm-up steps, and then decays it using a cosine function. This approach helps stabilize training in the early stages and allows for better convergence. Early stopping is also employed, where the training process is halted if the validation loss does not improve for a two consecutive epochs, to prevent overfitting and save computational resources.

We train each model using annually updated rolling window, given the time-varying universe of stocks in 1. Specifically, we use a 8-year in-sample period, with 6 years for training and 2 years for validation, to predict the out-of-sample returns for the subsequent year. The first training period starts from January 1996 to December 2003, and the trained model is then used to predict out-of-sample returns for the subsequent year 2004. The model is then updated with the data from 2004, and used to predict out-of-sample returns for 2005. This process continues until August 2023, the last month of our dataset. Overall, the out-of-sample prediction period is from January 2004 to August 2023, which contains around 20 years of results. The daily dataset provides a large number of training samples, which is crucial for training such a large deep learning model, while we only collect the out-of-sample predictions at the end of each month for empirical analysis. Since the stochastic nature of the Vision Transformer model, we train each specific model for five times and calculate the average predictions, following Gu et al. (2020).

4 Empirical Results

4.1 Binary Classification: Up and Down

We begin with the simplest binary classification problem ($K = 2$). Following Jiang et al. (2023), the stock returns are simply classified into "up" and "down" classes based on whether the future return is positive or negative. Besides the simplicity and interpretability, one remarkable benefit of such binary classification is that the two classes are naturally more balanced in our dataset, which is to say, there are approximately 50% "up" labels and 50% "down" labels. This is crucial for the training of such deep learning or machine learning models, since imbalanced classes can lead to suboptimal performance, as the model may become biased towards the majority class (He and Garcia, 2009).

We focus on the prediction of 1-month ahead stock returns ($H = 21$ trading days), which is common in the literature of cross-section of stock returns studies. We consider four different model specifications: (1) Temporal Transformer (TF) model with only stock volatility surface as input, the lookback window is $L = 21$ days, covering the last month. Since the volatility surface data contains 10 maturities ($N_\tau = 10$) and 18 moneyness levels ($N_\delta = 18$), we call this model **TF180**; (2) Temporal Transformer (TF) model with both stock and index volatility surfaces as input, the lookback window is also $L = 21$ days, resulting in an input size of $2 \times 10 \times 18 = 360$ features per day, we call this model **TF360**; (3) Vision Transformer (ViT) model with only stock volatility surface as input, we call this model **ViT180**; (4) Channel Vision Transformer (Channel ViT) model with both stock and index volatility surfaces as input, we call this model **ViT360**.

Table 1 reports the portfolio performance of equal-weight and value-weight decile portfolios, including a long-short spread portfolio "10-1", formed on out-of-sample predicted "up" probability $\mathbb{P}(r_{i,t}^{21} > 0)$ in each month's end, with a holding period of one month. All the four model specifications generate significant positive returns for the long-short portfolio. Moreover, Figure 7 plot the decile portfolio returns, as well as two of the most prominent option-base predictors in the literature: the implied volatility spread between ATM call and put options (**IVSpread**) defined in Yan (2011), and the implied volatility skew betwen OTM put and ATM call options (**IVSkew**) defined in Xing et al. (2010). More details about these two predictors are provided in Appendix B.

As mentioned in Muravyev et al. (2025), the spreads of the two predictors stem from the decile one's (short-leg) underperformance, while the returns in higher deciles are relatively flat. This is consistent with the intuition that options market contains more information about the extreme negative returns (left tail) since investors are more likely to join the options market for hedging purposes. The decile portfolio returns from our four Transformer-based models present a similar pattern, where the long legs are relatively flat and close to the two spread predictors but the short legs generate lower returns. This indicates that our models are able to capture more information about the downside risk, excelling the performance of the two spread predictors.

It is difficult to interpret the Transformer-based models directly, due to the complex nonlinear structure. We attempt to interpret the model predictions through relating them to the known option-based and fundamental characteristics. Table 2 reports the univariate correlations between the models' predictions and a set of these characteristics. Following the standard practice in the cross-section studies of stock returns, the correlation matrix is calculated on each month, and then averaged over the test sample period. The results of four specifications show a similar pattern: the model predictions are positively correlated with the firm size log(ME) and negatively correlated with CAPM beta (β), as well as the implied volatility level (ATM call and put IV) and idiosyncratic volatility (Ivol). Table 3 further reports the portfolio characteristics of decile portfolios and presented similar patterns.

In addition to the standard portfolio analysis, we also examine the time series regression of the long-short value-weighted portfolio returns on common risk factors. Table 4 reports the results of regressing the long-short portfolio returns on the Fama-French 5 factors (Fama and French, 2015) augmented with the momentum factor (Carhart, 1997). All the four specifications generate significant positive alphas, ranging from 1.1% to 1.5% on a monthly basis, economically larger than the IVSpread and IVSkew predictors. The factor exposures are also similar among the four models, with significant negative loadings on the market factor (MKT) and size factor (SMB), and significant positive loadings on the profitability factor (RMW).

Table 5 further reports the time series regression results when using the option-based factors. The ViT360 model generates a significant alpha of 1.8% on monthly basis, while the other three models also generate positive alphas, though not statistically significant. The results of alphas, as well as the regression R^2, suggest that the ViT360 model is able to capture more information from the volatility surfaces beyond the known option-based predictors, while the predictive power of the other three models can be mostly explained by the known option characteristics.

The Fama-MacBeth regression results in Table 6 further confirm the incremental predictive power of our models. The ViT360 model generates a significant positive coefficient, even after controlling for a comprehensive set of option-based and fundamental characteristics. The other three models also generate positive coefficients, though not statistically significant after controlling for those characteristics.

Since the profit of the long-short portfolio mainly drives from the short leg (decile 1), which has the smallest average market size (Table 3), we further conduct the same analysis on the sample excluding the smallest 20% stocks according to the NYSE size breakpoints. The results are reported in Appendix C, and the conclusions remain unchanged.

Following Jiang et al. (2023), we conduct a logistic regression of the realized future return indicator on the out-of-sample forecasted probabilities, while controlling for the other option-based and fundamental characteristics. We focus on the best-performing ViT360 model, and Table 7 reports the results. The dependent variable is an indicator for a positive 1-month future return, and the positive and significant coefficient of our predicted probability indicates a strong predictive power, even after controlling for the previously studied characteristics. In Table 8, we further conduct a series of logistic regressions using the crash indicator as the dependent variable, defined as whether the future return is below certain negative thresholds (-10%, -20%, -30%). Consistent with the strong predictive power for extreme negative returns discussed above, the coefficients of our predicted probabilities are significantly negative and have larger economic magnitudes, compared the results when using the positive return indicator as the dependent variable.

4.2 Multi-Class Classification: Quantile Bins

In this section, we extend our analysis to multi-class classification problems ($K > 2$). Since the stock returns are continuous variables, we discretize them into multiple classes based on the quantiles of the unconditional stock return distribution, as described in Section 3.3.

For each class k, we approximately assume that the stock returns falling into class k follow a uniform distribution within the interval $(q_{k-1}, q_k]$. Therefore, the predicted probability $\hat{y}_{i,t}^k$ can be interpreted as the likelihood of stock i's return falling into the interval $(q_{k-1}, q_k]$ at time t. Based on this interpretation, we can estimate the moment forecasts of the stock returns using the predicted probabilities from the multi-class classification model.

$$\mathbb{E}(r_{i,t}^{H}) = \sum_{k=1}^{K} \hat{y}_{i,t}^{k} \cdot \bar{q}_k \tag{17}$$

$$\mathrm{Var}(r_{i,t}^{H}) = \sum_{k=1}^{K} \hat{y}_{i,t}^{k} \cdot \left(\bar{q}_k - \mathbb{E}(r_{i,t}^{H})\right)^2 \tag{18}$$

$$\mathrm{Skew}(r_{i,t}^{H}) = \sum_{k=1}^{K} \hat{y}_{i,t}^{k} \cdot \left(\bar{q}_k - \mathbb{E}(r_{i,t}^{H})\right)^3 / \left(\mathrm{Var}(r_{i,t}^{H})\right)^{3/2} \tag{19}$$

$$\mathrm{Kurt}(r_{i,t}^{H}) = \sum_{k=1}^{K} \hat{y}_{i,t}^{k} \cdot \left(\bar{q}_k - \mathbb{E}(r_{i,t}^{H})\right)^4 / \left(\mathrm{Var}(r_{i,t}^{H})\right)^{2} \tag{20}$$

Where $\bar{q}_k = (q_{k-1} + q_k)/2$ is the midpoint of the interval $(q_{k-1}, q_k]$. The estimated $\mathbb{E}(r_{i,t}^{H})$, $\mathrm{Var}(r_{i,t}^{H})$, $\mathrm{Skew}(r_{i,t}^{H})$, and $\mathrm{Kurt}(r_{i,t}^{H})$ represent the predicted mean, variance, skewness, and kurtosis of stock i's return from day t to day $t+H$, respectively.

Figure 9 shows that average portfolio returns sorted by the predicted mean return $\mathbb{E}(r_{i,t}^{21})$ are broadly increasing with the portfolio's average predicted mean return $\mathbb{E}(r_{i,t}^{21})$, and the predicted mean return captures significant cross-sectional variation in stock returns, with a R-squared ranging from 77.1% to 93.7%. In this section, we focus on the **ViT360** model, and select $K = 100$ classes (percentile bins) for the multi-class classification, which performs the best compared to other specification like **TF360** (see Figure C.2), and other choices of K (see Figure C.1). We also report the results using other characteristic-sorted portfolios in Figure C.3 as a preliminary check.

4.3 Multi-Class Classification: Joint Distribution of Stock and Market Returns

In this section, we further explore the capability of our Transformer-based models to recover the joint distribution of individual stock returns and market returns.

The first challenge we face is the discretization of the joint distribution. Since the individual stock returns are positively correlated with the market returns in general, we cannot simply discretize the two returns independently, which would lead to a large number of empty or sparse joint classes. Instead, we consider two methods to mitigate this issues. First, we use a popular unsupervised learning algorithm, K-means clustering (McQueen, 1967), to group the joint return observations into K clusters based on their Euclidean distances in the two-dimensional return space. Each cluster represents a joint class, and the centroids of the clusters can be used as representative points for each class. Figure 10 shows the clustering results with $K = 100$ clusters. Second, even the K-Means clustering cannot ensure a fully balanced class distribution, we need to use the weighted cross-entropy loss function described in Section 3.4 to address the class imbalance issue during training.

For each class k, we denote the centroid of the cluster as $(\bar{r}_s^k, \bar{r}_m^k)$. Similar to the previous section, we can estimate the moments of both individual stock returns and market returns using the predicted probabilities $\hat{y}_{i,t}^k$ from the multi-class classification model. A important benefit of this approach is that we can also estimate the predicted covariance between the individual stock returns and market returns:

$$\mathrm{Cov}(r_{i,t}^H, r_{m,t}^H) = \sum_{k=1}^{K} \hat{y}_{i,t}^k \cdot \left(\bar{r}_s^k - \mathbb{E}(r_{i,t}^H)\right) \cdot \left(\bar{r}_m^k - \mathbb{E}(r_{m,t}^H)\right) \tag{21}$$

Furthermore, we can estimate the predicted Capital Asset Pricing Model (CAPM) beta of individual stocks using the predicted covariance and variance:

$$\beta_{i,t}^{fwd} = \frac{\mathrm{Cov}(r_{i,t}^H, r_{m,t}^H)}{\mathrm{Var}(r_{m,t}^H)} \tag{22}$$

Expected stock return conditional on the market crash can be estimated as:

$$\mathbb{E}(r_{i,t}^H | r_{m,t}^H < q_{crash}) = \frac{\sum_{k:\bar{r}_m^k < q_{crash}} \hat{y}_{i,t}^k \cdot \bar{r}_s^k}{\sum_{k:\bar{r}_m^k < q_{crash}} \hat{y}_{i,t}^k} \tag{23}$$

5 Conclusion

References

Amihud, Y., 2002, “Illiquidity and Stock Returns: Cross-Section and Time-Series Effects,” *Journal of Financial Markets*, 5 (1), 31–56, January.

An, B.-J., A. Ang, T. G. Bali, and N. Cakici, 2014, “The Joint Cross Section of Stocks and Options,” *The Journal of Finance*, 69 (5), 2279–2337.

Andersen, T. G., N. Fusari, and V. Todorov, 2015, “The Risk Premia Embedded in Index Options,” *Journal of Financial Economics*, 117 (3), 558–584, September.

Ang, A., R. J. Hodrick, Y. Xing, and X. Zhang, 2006, “The Cross-Section of Volatility and Expected Returns,” *The Journal of Finance*, 61 (1), 259–299.

Arnab, A., M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, 2021, “ViViT: A Video Vision Transformer,” November.

Bali, T. G. and A. Hovakimian, 2009, “Volatility Spreads and Expected Stock Returns.,” *Management Science*, 55 (11), 1797–1812, November.

Ball, R., J. Gerakos, J. T. Linnainmaa, and V. Nikolaev, 2016, “Accruals, Cash Flows, and Operating Profitability in the Cross Section of Stock Returns,” *Journal of Financial Economics*, 121 (1), 28–45, July.

Baltussen, G., S. van Bekkum, and B. van der Grient, 2018, “Unknown Unknowns: Uncertainty About Risk and Stock Returns.,” *Journal of Financial & Quantitative Analysis*, 53 (4), 1615–1651, August.

Bao, Y., S. Sivanandan, and T. Karaletsos, 2024, “Channel Vision Transformers: An Image Is Worth 1 x 16 x 16 Words,” April.

Bertasius, G., H. Wang, and L. Torresani, 2021, “Is Space-Time Attention All You Need for Video Understanding?,” June.

Beyer, L., X. Zhai, and A. Kolesnikov, 2022, “Better Plain ViT Baselines for ImageNet-1k,” May.

Bianchi, D., M. Büchner, and A. Tamoni, 2021, “Bond Risk Premiums with Machine Learning,” *The Review of Financial Studies*, 34 (2), 1046–1089, January.

Breeden, D. T. and R. H. Litzenberger, 1978, “Prices of State-Contingent Claims Implicit in Option Prices,” *The Journal of Business*, 51 (4), 621–651.

Brown, T. B., B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, 2020, “Language Models Are Few-Shot Learners,” July.

Carhart, M. M., 1997, “On Persistence in Mutual Fund Performance,” *The Journal of Fi-*

nance, 52 (1), 57–82.

Chen, A. Y. and T. Zimmermann, 2022, "Open Source Cross-Sectional Asset Pricing," *Critical Finance Review*, 11 (2), 207–264.

Cooper, M. J., H. Gulen, and M. J. Schill, 2008, "Asset Growth and the Cross-Section of Stock Returns," *The Journal of Finance*, 63 (4), 1609–1651.

Cox, J. C. and S. A. Ross, 1976a, "A Survey of Some New Results in Financial Option Pricing Theory," *The Journal of Finance*, 31 (2), 383–402.

Cox, J. C. and S. A. Ross, 1976b, "The Valuation of Options for Alternative Stochastic Processes," *Journal of Financial Economics*, 3 (1), 145–166, January.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova, 2019, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in J. Burstein, C. Doran, and T. Solorio eds. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics, June.

Dosovitskiy, A., L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, and S. Gelly, 2020, "An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale," *arXiv preprint arXiv:2010.11929*.

Fama, E. F. and K. R. French, 1992, "The Cross-Section of Expected Stock Returns," *The Journal of Finance*, 47 (2), 427–465, June.

Fama, E. F. and K. R. French, 2015, "A Five-Factor Asset Pricing Model," *Journal of Financial Economics*, 116 (1), 1–22, April.

Fama, E. F. and J. D. MacBeth, 1973, "Risk, Return, and Equilibrium: Empirical Tests.," *Journal of Political Economy*, 81 (3), p. 607, May.

Glorot, X. and Y. Bengio, 2010, "Understanding the Difficulty of Training Deep Feedforward Neural Networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 249–256. JMLR Workshop and Conference Proceedings, March.

Gu, S., B. Kelly, and D. Xiu, 2020, "Empirical Asset Pricing via Machine Learning.," *Review of Financial Studies*, 33 (5), 2223–2273.

He, H. and E. A. Garcia, 2009, "Learning from Imbalanced Data," *IEEE Transactions on Knowledge and Data Engineering*, 21 (9), 1263–1284, September.

Hendrycks, D. and K. Gimpel, 2023, "Gaussian Error Linear Units (GELUs)," June.

Jegadeesh, N., 1990, "Evidence of Predictable Behavior of Security Returns," *The Journal of Finance*, 45 (3), 881–898.

Jegadeesh, N. and S. Titman, 1993, "Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency," *The Journal of Finance*, 48 (1), 65–91.

Jiang, J., B. Kelly, and D. Xiu, 2023, "(Re-)Imag(in)Ing Price Trends," *The Journal of Finance*, 78 (6), 3193–3249.

Johnson, T. L. and E. C. So, 2012, "The Option to Stock Volume Ratio and Future Returns," *Journal of Financial Economics*, 106 (2), 262–286, November.

Kelly, B. T., B. Kuznetsov, S. Malamud, and T. A. Xu, 2023, "Deep Learning from Implied Volatility Surfaces," August.

Kingma, D. P. and J. Ba, 2017, "Adam: A Method for Stochastic Optimization," January.

Loshchilov, I. and F. Hutter, 2019, "Decoupled Weight Decay Regularization," January.

Martin, I., 2017, "What Is the Expected Return on the Market?*," *The Quarterly Journal of Economics*, 132 (1), 367–433, February.

Martin, I. W. R. and C. Wagner, 2019, "What Is the Expected Return on a Stock?" *The Journal of Finance*, 74 (4), 1887–1929.

McQueen, J. B., 1967, "Some Methods of Classification and Analysis of Multivariate Observations," in *Proc. of 5th Berkeley Symposium on Math. Stat. and Prob.*, 281–297.

Muravyev, D., N. D. Pearson, and J. M. Pollet, 2025, "Why Does Options Market Information Predict Stock Returns?" *Journal of Financial Economics*, 172, p. 104153, October.

Neuhierl, A., X. Tang, R. T. Varneskov, and G. Zhou, 2022, "Option Characteristics as Cross-Sectional Predictors."

Newey, W. K. and K. D. West, 1987, "A Simple, Positive Semi-Definite, Heteroskedasticity and Autocorrelation Consistent Covariance Matrix," *Econometrica (1986-1998)*, 55 (3), p. 703, May.

Ross, S., 2015, "The Recovery Theorem," *The Journal of Finance*, 70 (2), 615–648.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, 2017, "Attention Is All You Need," *Advances in neural information processing systems*, 30.

Xing, Y., X. Zhang, and R. Zhao, 2010, "What Does the Individual Option Volatility Smirk Tell Us About Future Equity Returns?" *Journal of Financial and Quantitative Analysis*, 45 (3), 641–662, June.

Xiong, R., Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T.-Y. Liu, 2020, "On Layer Normalization in the Transformer Architecture," June.

Yan, S., 2011, "Jump Risk, Stock Returns, and Slope of Implied Volatility Smile," *Journal of Financial Economics*, 99 (1), 216–233, January.

Tables

Table 1: Portfolio Analysis, Binary Classification Model

	Decile Portfolios										
Model	1	2	3	4	5	6	7	8	9	10	10 - 1
					Equal-Weighted Portfolios						
TF180	-0.66	0.36	0.68	0.88*	0.90**	0.92**	0.86**	0.85***	0.80***	0.68***	1.33***
	(-0.99)	(0.57)	(1.24)	(1.80)	(2.10)	(2.36)	(2.39)	(2.69)	(2.89)	(2.97)	(2.70)
TF360	-0.51	0.12	0.55	0.71	0.84**	0.81**	0.91**	0.95***	0.99***	0.90**	1.41***
	(-0.80)	(0.21)	(1.08)	(1.54)	(2.08)	(2.13)	(2.55)	(2.74)	(2.71)	(1.96)	(2.83)
ViT180	-0.73	0.15	0.66	0.85*	0.88**	0.88**	0.84**	0.81***	0.74***	0.67***	1.39**
	(-0.98)	(0.25)	(1.25)	(1.89)	(2.19)	(2.40)	(2.47)	(2.63)	(2.81)	(3.12)	(2.37)
ViT360	-0.65	-0.08	0.35	0.73*	0.93**	0.99**	1.07***	1.01**	0.77**	0.65*	1.30***
	(-1.23)	(-0.17)	(0.76)	(1.70)	(2.27)	(2.49)	(2.60)	(2.51)	(2.04)	(1.82)	(3.45)
					Value-Weighted Portfolios						
TF180	-0.50	0.32	0.66	0.87*	0.96**	0.80**	0.92***	0.98***	0.84***	0.80***	1.30**
	(-0.84)	(0.46)	(1.12)	(1.80)	(2.16)	(2.03)	(2.62)	(3.31)	(3.35)	(4.10)	(2.54)
TF360	-0.52	0.25	0.55	0.74	0.60	0.78**	0.81**	1.09***	1.05***	1.11***	1.63***
	(-0.90)	(0.37)	(1.07)	(1.49)	(1.36)	(2.18)	(2.46)	(3.26)	(3.42)	(2.70)	(3.20)
ViT180	-0.60	0.24	0.66	0.86	0.70	0.89**	0.80**	0.88***	0.82***	0.71***	1.31**
	(-0.86)	(0.35)	(1.11)	(1.64)	(1.48)	(2.41)	(2.53)	(2.93)	(3.11)	(3.51)	(2.19)
ViT360	-0.59	-0.01	0.55	0.81*	0.94**	0.92**	1.01***	0.96**	0.83**	0.67*	1.27***
	(-1.08)	(-0.02)	(1.20)	(1.89)	(2.46)	(2.52)	(2.69)	(2.42)	(2.29)	(1.78)	(2.94)

This table reports the monthly average returns of equal-weight (top panel) and value-weight (bottom panel) decile portfolios formed on the out-of-sample predicted "up" probability $\mathbb{P}(r_{i,t}^{21} > 0)$ from four different model specifications described in Section 4.1. The long-short portfolio $(10-1)$ is formed by longing the highest decile and shorting the lowest decile. The portfolios are rebalanced monthly and held for one month. The sample period is from January 2004 to August 2023. The t-statistics are calculated using Newey and West (1987) adjusted standard errors with 6 lags, and reported in parentheses. ***, **, and * denote statistical significance at the 1%, 5%, and 10% levels, respectively.

Table 2: Correlation Analysis, Binary Classification Model, All Stocks

	TF180	TF360	ViT180	ViT360
CIV	−0.504	−0.401	−0.650	−0.457
PIV	−0.539	−0.419	−0.660	−0.468
SKEW	0.099	0.078	0.162	0.078
ΔCIV	0.006	0.008	−0.043	−0.026
ΔPIV	−0.003	0.001	−0.047	−0.029
log(O/S)	−0.021	−0.027	0.029	0.014
IVSpread	−0.089	−0.074	−0.065	−0.053
IVSkew	−0.011	−0.057	−0.040	−0.087
IVol-RVol	0.154	0.125	0.161	0.129
VOV	−0.081	−0.042	−0.144	−0.118
beta	−0.199	−0.232	−0.276	−0.185
log(ME)	0.428	0.306	0.513	0.373
log(B/M)	0.018	0.022	0.023	0.022
Inv	−0.057	−0.048	−0.058	−0.040
Profit	0.099	0.065	0.147	0.109
Illiq	−0.214	−0.167	−0.291	−0.214
Ivol	−0.548	−0.432	−0.645	−0.438
$ret_{12,2}$	0.108	0.085	0.135	0.056
$ret_{1,0}$	0.064	0.051	0.099	0.061

This table reports the average monthly cross-sectional correlations between the out-of-sample predicted "up" probabilities from four different model specifications described in Section 4.1 and a set of option-based and fundamental stock characteristics. The correlations are calculated on each month, and then averaged over the test sample period from January 2004 to August 2023. The stock characteristics are described in Appendix B.

Table 3: Portfolio Characteristics, Binary Classification Model, All Stocks

	Low	2	3	4	5	6	7	8	9	High
$\mathbb{P}(r_{i,t}^{21} > 0)$	0.448	0.505	0.541	0.565	0.582	0.595	0.606	0.615	0.625	0.646
CIV	1.291	0.843	0.685	0.587	0.516	0.454	0.414	0.368	0.336	0.384
PIV	1.174	0.787	0.651	0.560	0.491	0.432	0.394	0.352	0.321	0.369
SKEW	-0.088	-0.051	-0.011	0.003	-0.004	-0.006	-0.003	0.002	0.004	-0.009
ΔCIV	0.012	0.001	-0.000	-0.000	-0.001	-0.002	-0.002	-0.002	-0.002	-0.003
ΔPIV	0.010	0.002	0.000	0.000	0.000	-0.002	-0.003	-0.002	-0.002	-0.003
log(O/S)	5.388	5.554	5.725	5.747	5.685	5.605	5.584	5.617	5.640	5.384
IVSpread	0.021	0.009	0.005	0.004	0.004	0.001	-0.001	-0.001	0.001	0.002
IVSkew	0.109	0.084	0.073	0.072	0.071	0.070	0.068	0.066	0.068	0.087
IVol-RVol	-0.102	-0.061	-0.041	-0.033	-0.029	-0.025	-0.022	-0.018	-0.020	-0.027
VOV	0.217	0.179	0.151	0.145	0.144	0.144	0.142	0.141	0.143	0.175
beta	1.203	1.262	1.278	1.231	1.169	1.109	1.046	0.970	0.879	0.834
log(ME)	12.502	13.080	13.499	13.821	14.096	14.347	14.618	14.860	14.969	14.443
log(B/M)	1.068	1.117	1.121	1.136	1.221	1.270	1.264	1.296	1.205	1.252
Inv	0.182	0.193	0.200	0.183	0.167	0.154	0.136	0.127	0.115	0.114
Profit	0.013	0.049	0.072	0.085	0.088	0.086	0.080	0.072	0.058	0.042
Illiq	0.088	0.053	0.035	0.025	0.022	0.018	0.017	0.014	0.014	0.025
Ivol	0.032	0.026	0.023	0.020	0.018	0.016	0.015	0.014	0.012	0.013
$ret_{12,2}$	0.081	0.166	0.188	0.160	0.143	0.130	0.132	0.123	0.118	0.108
$ret_{1,0}$	-0.945	-0.149	0.431	0.821	0.999	1.096	1.185	1.137	1.053	0.883

This table report the time series average characteristics of ten equal-weight decile portfolios formed on the out-of-sample predicted "up" probability from model **ViT360** described in Section 4.1. The sample period is from January 2004 to August 2023. The stock characteristics are described in Appendix B.

Table 4: Time Series Regression, Binary Classification Model, All Stocks

	TF180	TF360	ViT180	ViT360	IVSpread	IVSkew
	(1)	(2)	(3)	(4)	(5)	(6)
Alpha	0.011***	0.015***	0.012***	0.011***	0.008***	0.001
	(0.003)	(0.004)	(0.004)	(0.003)	(0.002)	(0.002)
MKT	-0.403***	-0.475***	-0.602***	-0.277**	-0.021	0.051
	(0.095)	(0.114)	(0.062)	(0.115)	(0.052)	(0.057)
SMB	-0.924***	-0.942***	-1.129***	-0.898***	-0.024	-0.156
	(0.186)	(0.182)	(0.146)	(0.180)	(0.083)	(0.115)
HML	0.393***	0.466**	0.312**	0.291	0.186*	-0.311***
	(0.120)	(0.184)	(0.145)	(0.286)	(0.110)	(0.089)
RMW	1.224***	1.187***	1.566***	0.971***	0.378***	-0.056
	(0.187)	(0.233)	(0.204)	(0.263)	(0.117)	(0.155)
CMA	-0.029	-0.305	0.302	-0.423	0.032	-0.334**
	(0.231)	(0.338)	(0.275)	(0.386)	(0.195)	(0.153)
UMD	0.146	-0.282	0.344*	-0.116	0.137***	0.163**
	(0.228)	(0.262)	(0.185)	(0.182)	(0.040)	(0.081)
Observations	220	220	236	236	228	228
R^2	0.534	0.446	0.668	0.279	0.130	0.277
Adjusted R^2	0.521	0.431	0.659	0.260	0.106	0.257
Residual Std. Error	0.047	0.054	0.049	0.062	0.029	0.030
F Statistic	21.175***	21.231***	88.988***	8.865***	6.669***	8.560***

This table reports the alpha and factor loadings from the time series regression of the long-short $(10-1)$ value-weighted portfolio returns on the Fama-French 5 factors (Fama and French, 2015) augmented with the momentum factor (Carhart, 1997). The portfolio is formed on the out-of-sample predicted "up" probability $\mathbb{P}(r_{i,t}^{21} > 0)$ from four different model specifications described in Section 4.1. The sample period is from January 2004 to August 2023. The t-statistics are calculated using Newey and West (1987) adjusted standard errors with 6 lags, and reported in parentheses. ***, **, and * denote statistical significance at the 1%, 5%, and 10% levels, respectively.

Table 5: Time Series Regression on Option-Based Factors

	TF180	TF360	ViT180	ViT360
	(1)	(2)	(3)	(4)
Alpha	0.006	0.007	0.001	0.018***
	(0.006)	(0.007)	(0.007)	(0.007)
CIV	-0.827	0.685	-0.846	1.089*
	(0.590)	(0.622)	(0.840)	(0.656)
PIV	0.836	-0.658	0.899	-1.127*
	(0.577)	(0.598)	(0.838)	(0.655)
ΔCIV	-0.011***	-0.008***	-0.016***	-0.008**
	(0.003)	(0.003)	(0.004)	(0.003)
ΔPIV	-0.001	-0.000	-0.003	-0.003
	(0.002)	(0.003)	(0.003)	(0.004)
SKEW	-0.357	0.124	-0.215	0.060
	(0.345)	(0.165)	(0.301)	(0.200)
log(O/S)	0.003**	0.004**	0.003	0.002
	(0.002)	(0.002)	(0.002)	(0.002)
IVSpread	0.005***	0.008***	0.010***	-0.001
	(0.002)	(0.003)	(0.002)	(0.003)
IVSkew	-0.002	-0.004*	-0.003	-0.002
	(0.003)	(0.003)	(0.003)	(0.003)
VOV	0.516**	-0.032	0.875***	0.456**
	(0.248)	(0.238)	(0.299)	(0.180)
IVol-RVol	-0.002	-0.004***	-0.002	-0.001
	(0.001)	(0.001)	(0.002)	(0.001)
Observations	213	213	228	228
R^2	0.242	0.362	0.314	0.096
Adjusted R^2	0.205	0.331	0.283	0.054
Residual Std. Error	0.062	0.059	0.071	0.069
F Statistic	6.111***	19.348***	15.803***	3.081***

This table reports the alpha and factor loadings from the time series regression of the long-short $(10-1)$ value-weighted portfolio returns on a set of option-based factors described in Appendix B. The portfolio is formed on the out-of-sample predicted "up" probability $\mathbb{P}(r_{i,t}^{21} > 0)$ from four different model specifications described in Section 4.1. The sample period is from January 2004 to August 2023. The t-statistics are calculated using Newey and West (1987) adjusted standard errors with 6 lags, and reported in parentheses. ***, **, and * denote statistical significance at the 1%, 5%, and 10% levels, respectively.

Table 6: Fama-MacBeth Regression, Binary Classification Model, All Stocks

	TF180		TF360		ViT180		ViT360	
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
$\mathbb{P}(r_{i,t}^{21} > 0)$	0.082**	-0.007	0.115**	0.061	0.103**	0.059*	0.075***	0.049**
	(0.034)	(0.027)	(0.052)	(0.056)	(0.044)	(0.032)	(0.021)	(0.023)
CIV		-0.034***		-0.030***		-0.031***		-0.027***
		(0.009)		(0.009)		(0.008)		(0.007)
PIV		0.019**		0.019**		0.023**		0.024**
		(0.009)		(0.009)		(0.009)		(0.010)
SKEW		-0.010		-0.009		-0.011		-0.008
		(0.007)		(0.007)		(0.007)		(0.007)
ΔCIV		-0.006		-0.007		-0.008		-0.007
		(0.007)		(0.007)		(0.007)		(0.008)
ΔPIV		0.018**		0.017**		0.015*		0.014*
		(0.008)		(0.008)		(0.008)		(0.008)
log(O/S)		-0.000		-0.000		0.000		-0.000
		(0.000)		(0.000)		(0.000)		(0.000)
IVSpread		-0.065***		-0.063***		-0.064***		-0.061***
		(0.012)		(0.012)		(0.013)		(0.012)
IVSkew		-0.009		-0.009		-0.011**		-0.009*
		(0.006)		(0.006)		(0.005)		(0.005)
IVol-RVol		0.005		0.005		0.003		0.003
		(0.004)		(0.004)		(0.003)		(0.003)
beta		0.002		0.002		0.002		0.002
		(0.003)		(0.003)		(0.003)		(0.003)
log(ME)		-0.001		-0.001		-0.001		-0.000
		(0.001)		(0.001)		(0.001)		(0.000)
N	836,275	321,721	836,275	321,721	904,497	344,223	904,497	344,223
Adj R^2	0.022	0.070	0.022	0.070	0.030	0.068	0.021	0.068

This table reports the Fama and MacBeth (1973) regression results of future 1-month stock returns on the out-of-sample predicted "up" probabilities from four different model specifications described in Section 4.1, along with a comprehensive set of option-based and fundamental stock characteristics described in Appendix B. The sample period is from January 2004 to August 2023. The t-statistics are calculated using Newey and West (1987) adjusted standard errors with 6 lags, and reported in parentheses. ***, **, and * denote statistical significance at the 1%, 5%, and 10% levels, respectively.

Table 7: Logistic Regression, Future Stock Return

	Dependent variable: $\mathbf{I}(r_{i,t}^{21} > 0)$		
	(1)	(2)	(3)
$\mathbb{P}(r_{i,t}^{21} > 0)$	0.616***		0.391***
	(0.015)		(0.025)
CIV		-0.172***	-0.183***
		(0.057)	(0.057)
PIV		-0.147**	-0.188***
		(0.062)	(0.062)
SKEW		0.193***	0.211***
		(0.041)	(0.041)
ΔCIV		0.090*	0.114**
		(0.049)	(0.049)
ΔPIV		-0.057	-0.057
		(0.049)	(0.049)
log(O/S)		-0.029***	-0.026***
		(0.003)	(0.003)
IVSpread		-0.403***	-0.368***
		(0.069)	(0.069)
IVSkew		-0.156***	-0.165***
		(0.046)	(0.046)
IVol-RVol		0.035*	0.023
		(0.020)	(0.020)
VOV		0.056	0.086*
		(0.048)	(0.048)
beta		-0.021**	-0.007
		(0.008)	(0.008)
log(ME)		0.031***	0.023***
		(0.003)	(0.003)
$ret_{12,2}$		-0.012***	-0.006
		(0.004)	(0.004)
$ret_{1,0}$		-0.001***	-0.001***
		(0.000)	(0.000)
Observations	904497	342535	342535
Pseudo R^2	0.001	0.003	0.003

This table reports the panel logistic regression results of future return indicator on the out-of-sample predicted "up" probabilities from model **ViT360** described in Section 4.1, along with a comprehensive set of option-based and fundamental stock characteristics described in Appendix B. The dependent variable is an indicator for a positive 1-month future return. The test sample period is from January 2004 to August 2023. The t-statistics are calculated using Newey and West (1987) adjusted standard errors with 6 lags, and reported in parentheses. ***, **, and * denote statistical significance at the 1%, 5%, and 10% levels, respectively.

Table 8: Logistic Regression, Future Stock Crash

	$\mathbf{I}(r_{i,t}^H < -10\%)$			$\mathbf{I}(r_{i,t}^H < -20\%)$			$\mathbf{I}(r_{i,t}^H < -30\%)$		
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
$\mathbb{P}(r_{i,t}^{21} > 0)$	-1.411***		-0.673***	-1.974***		-1.341***	-2.042***		-1.529***
	(0.031)		(0.038)	(0.056)		(0.071)	(0.089)		(0.119)
CIV		0.546***	0.559***		0.573***	0.607***		0.722***	0.773***
		(0.125)	(0.128)		(0.149)	(0.157)		(0.125)	(0.130)
PIV		1.485***	1.558***		1.956***	2.072***		1.551***	1.646***
		(0.133)	(0.137)		(0.164)	(0.173)		(0.152)	(0.156)
SKEW		0.070	0.030		0.141	0.052		0.126	0.030
		(0.064)	(0.065)		(0.101)	(0.103)		(0.120)	(0.122)
ΔCIV		-0.018	-0.065		0.201*	0.120		0.531***	0.454***
		(0.069)	(0.069)		(0.107)	(0.107)		(0.160)	(0.161)
ΔPIV		0.210***	0.217***		0.409***	0.413***		0.467***	0.458***
		(0.069)	(0.069)		(0.105)	(0.106)		(0.157)	(0.160)
log(O/S)		0.093***	0.086***		0.153***	0.140***		0.202***	0.187***
		(0.005)	(0.005)		(0.009)	(0.009)		(0.015)	(0.015)
IVSpread		0.130	0.067		0.269*	0.186		0.500***	0.447**
		(0.109)	(0.111)		(0.142)	(0.145)		(0.178)	(0.183)
IVSkew		-0.109	-0.100		-0.188*	-0.188		0.142	0.139
		(0.068)	(0.069)		(0.113)	(0.115)		(0.171)	(0.174)
IVol-RVol		0.162***	0.182***		0.315***	0.353***		0.325***	0.365***
		(0.028)	(0.029)		(0.046)	(0.048)		(0.045)	(0.045)
VOV		-1.110***	-1.150***		-1.331***	-1.385***		-0.909***	-0.950***
		(0.081)	(0.081)		(0.155)	(0.157)		(0.245)	(0.248)
beta		0.195***	0.175***		0.261***	0.230***		0.307***	0.282***
		(0.012)	(0.012)		(0.020)	(0.021)		(0.033)	(0.034)
log(ME)		-0.131***	-0.115***		-0.210***	-0.175***		-0.253***	-0.212***
		(0.005)	(0.005)		(0.008)	(0.009)		(0.014)	(0.015)
$ret_{12,2}$		-0.014***	-0.023***		-0.020*	-0.036***		-0.054**	-0.072***
		(0.005)	(0.006)		(0.011)	(0.013)		(0.022)	(0.022)
$ret_{1,0}$		-0.004***	-0.004***		-0.007***	-0.007***		-0.011***	-0.011***
		(0.000)	(0.000)		(0.001)	(0.001)		(0.001)	(0.001)
Observations	904497	342535	342535	904497	342535	342535	904497	342535	342535
Pseudo R^2	0.005	0.062	0.063	0.008	0.119	0.122	0.007	0.141	0.145

This table reports the panel logistic regression results of future crash indicators on the out-of-sample predicted "up" probabilities from model **ViT360** described in Section 4.1, along with a comprehensive set of option-based and fundamental stock characteristics described in Appendix B. The dependent variables are indicators for future 1-month returns below certain negative thresholds (-10%, -20%, -30%). The test sample period is from January 2004 to August 2023. The t-statistics are calculated using Newey and West (1987) adjusted standard errors with 6 lags, and reported in parentheses. ***, **, and * denote statistical significance at the 1%, 5%, and 10% levels, respectively.

Table 9: Summary Statistics of Predicted Moments, Quantile100 Classification

	N	Mean	Std	Skew	Min	25th	Median	75th	Max
$\mathbb{E}(r_{i,t}^{21})$	3833	0.008	0.011	-1.222	-0.040	0.005	0.010	0.014	0.044
$\mathrm{Var}(r_{i,t}^{21})$	3833	0.024	0.021	1.876	0.004	0.009	0.016	0.032	0.129
Anualized $\mathrm{Std}(r_{i,t}^{21})$	3833	0.480	0.202	1.041	0.219	0.322	0.427	0.599	1.214
$\mathrm{Skew}(r_{i,t}^{21})$	3833	0.239	0.270	1.631	-0.316	0.083	0.192	0.338	1.984
$\mathrm{Kurt}(r_{i,t}^{21})$	3833	8.704	7.210	2.609	0.726	4.290	6.534	10.551	47.946
Realized Return	3833	0.007	0.131	2.246	-0.799	-0.050	0.004	0.057	2.022
Realized Volatility	3831	0.410	0.293	5.092	0.006	0.238	0.342	0.501	5.468

This table provides summary statistics for the out-of-sample predicted 1-month stock returns moments described in Section 4.2, using the **ViT360** model with 100 quantile classes. The sample period is from January 2004 to August 2023. At the end of each month, we calculate the cross-sectional statistics of the predicted moments, and then report the time series averages of these statistics. We also report the realized stock return and realized volatility for comparison.

Table 10: Time Series Regression, Quantile100 Classification

	CAPM	FF3	FFC	FF5	FF6	Q5
	(1)	(2)	(3)	(4)	(5)	(6)
Alpha	0.013***	0.012***	0.014***	0.011***	0.012***	0.013***
	(0.005)	(0.004)	(0.004)	(0.004)	(0.004)	(0.004)
MKT	-0.091	0.090	-0.094	-0.007	-0.143	
	(0.253)	(0.212)	(0.168)	(0.168)	(0.143)	
SMB		-0.984***	-1.031***	-0.831***	-0.865***	
		(0.237)	(0.205)	(0.203)	(0.198)	
HML		0.085	-0.131	0.725**	0.454*	
		(0.204)	(0.158)	(0.284)	(0.241)	
RMW				0.654**	0.650*	
				(0.325)	(0.333)	
CMA				-1.397***	-1.152***	
				(0.418)	(0.361)	
UMD			-0.581***		-0.500***	
			(0.199)		(0.186)	
R_MKT						0.076
						(0.195)
R_ME						-0.796***
						(0.206)
R_IA						-0.150
						(0.341)
R_ROE						-0.295
						(0.559)
R_EG						0.268
						(0.562)
Observations	236	236	236	236	236	236
R^2	0.003	0.107	0.214	0.215	0.291	0.097
Adjusted R^2	-0.001	0.096	0.200	0.198	0.272	0.077
Residual Std. Error	0.070	0.066	0.062	0.063	0.060	0.067
F Statistic	0.130	9.581***	13.097***	8.357***	11.871***	11.078***

Note: *p<0.1; **p<0.05; ***p<0.01

This table reports the alpha and factor loadings from the time series regression of the long-short $(10-1)$ value-weighted portfolio returns sorted by the predicted mean return $\mathbb{E}(r_{i,t}^{21})$ described in Section 4.2, using the **ViT360** model with 100 quantile classes. The sample period is from January 2004 to August 2023. The t-statistics are calculated using Newey and West (1987) adjusted standard errors with 6 lags, and reported in parentheses. ***, **, and * denote statistical significance at the 1%, 5%, and 10% levels, respectively.

Table 11: Fama-MacBeth Regression, Quantile100 Classification

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
$\mathbb{E}(r_{i,t}^{21})$		0.779***	0.407**						
		(0.177)	(0.192)						
$\mathrm{Var}(r_{i,t}^{21})$				-0.265***	-0.267**				
				(0.086)	(0.109)				
$\mathrm{Skew}(r_{i,t}^{21})$						-0.005	0.010		
						(0.006)	(0.007)		
$\mathrm{Kurt}(r_{i,t}^{21})$								0.000	-0.001*
								(0.000)	(0.001)
CIV	-0.026***		-0.019**		-0.018**		-0.025***		-0.028***
	(0.009)		(0.008)		(0.008)		(0.009)		(0.010)
PIV	0.011		0.011		0.017		0.008		0.007
	(0.010)		(0.010)		(0.010)		(0.009)		(0.008)
SKEW	-0.004		-0.005		-0.004		-0.004		-0.004
	(0.007)		(0.007)		(0.007)		(0.007)		(0.007)
ΔCIV	-0.009		-0.008		-0.010		-0.008		-0.007
	(0.007)		(0.007)		(0.007)		(0.006)		(0.006)
ΔPIV	0.017**		0.014*		0.013*		0.017**		0.017**
	(0.007)		(0.007)		(0.007)		(0.007)		(0.008)
log(O/S)	-0.000		-0.000		0.000		-0.000		0.000
	(0.000)		(0.000)		(0.000)		(0.000)		(0.000)
IVSpread	-0.056***		-0.050***		-0.053***		-0.053***		-0.054***
	(0.013)		(0.012)		(0.012)		(0.012)		(0.012)
IVSkew	-0.006		-0.005		-0.005		-0.006		-0.007
	(0.005)		(0.005)		(0.005)		(0.005)		(0.005)
IVol-RVol	0.004		0.004		0.004		0.004		0.004
	(0.003)		(0.003)		(0.003)		(0.003)		(0.003)
beta	-0.000		-0.000		0.000		-0.000		-0.000
	(0.003)		(0.003)		(0.003)		(0.003)		(0.003)
log(ME)	-0.001*		-0.001		-0.001*		-0.001		-0.001*
	(0.000)		(0.000)		(0.000)		(0.000)		(0.000)
log(B/M)	-0.002		-0.002		-0.002		-0.002		-0.002
	(0.002)		(0.002)		(0.002)		(0.002)		(0.002)
$ret_{12,2}$	0.001		0.001		0.001		0.001		0.001
	(0.003)		(0.003)		(0.003)		(0.003)		(0.003)
$ret_{1,0}$	-0.000		-0.000		-0.000		-0.000		-0.000
	(0.000)		(0.000)		(0.000)		(0.000)		(0.000)
N	321,390	904,497	321,390	904,497	321,390	904,497	321,390	904,497	321,390
Adj R^2	0.089	0.021	0.091	0.032	0.092	0.010	0.091	0.019	0.092

This table reports the Fama and MacBeth (1973) regression results of future 1-month stock returns on the predicted stock return moments described in Section 4.2, using the **ViT360** model with 100 quantile classes. The sample period is from January 2004 to August 2023. The t-statistics are calculated using Newey and West (1987) adjusted standard errors with 6 lags, and reported in parentheses. ***, **, and * denote statistical significance at the 1%, 5%, and 10% levels, respectively.

Table 12: Summary Statistics of Predicted Moments, K-Means Clustering

	N	Mean	Std	Skew	Min	25th	Median	75th	Max
$\mathbb{E}(r_s)$	500	0.016	0.010	0.204	-0.011	0.010	0.015	0.021	0.047
Realized Return	500	0.009	0.074	0.322	-0.307	-0.032	0.008	0.049	0.413
Annualized Std(r_s)	500	0.503	0.091	1.883	0.389	0.441	0.475	0.536	0.923
Realized Volatility	500	0.293	0.137	2.447	0.054	0.205	0.263	0.345	1.305
Skew(r_s)	500	0.453	0.082	0.334	0.258	0.400	0.453	0.508	0.732
$\mathbb{E}(r_m)$	500	0.008	0.005	-0.592	-0.012	0.006	0.009	0.012	0.025
Annualized Std(r_m)	500	0.197	0.009	0.860	0.173	0.192	0.197	0.203	0.230
β^{hist}	498	1.029	0.385	0.537	0.163	0.767	0.994	1.257	2.510
β^{fwd}	500	1.021	0.116	0.543	0.762	0.946	1.005	1.082	1.426

This table provides summary statistics for the out-of-sample predicted 1-month stock and market returns joint distribution moments described in Section 4.3, using the **ViT360** model with K-Means clustering of 100 clusters. The sample period is from January 2004 to August 2023 and we use the stocks included in the S&P 500 index at each month-end. At the end of each month, we calculate the cross-sectional statistics of the predicted moments, and then report the time series averages of these statistics. We also report the realized stock return, realized volatility and historical CAPM beta for comparison.

Table 13: Cross-Sectional Regression, K-Means Clustering

	10 Portfolios		25 Portfolios		50 Portfolios	
	(1)	(2)	(3)	(4)	(5)	(6)
const	-0.002	0.007***	-0.002	0.007***	-0.000	0.007***
	(0.004)	(0.001)	(0.003)	(0.001)	(0.003)	(0.001)
β^{fwd}	0.010**		0.010***		0.008**	
	(0.004)		(0.003)		(0.003)	
β^{hist}		0.001		0.001		0.001
		(0.001)		(0.001)		(0.001)
R^2	0.493	0.128	0.311	0.055	0.127	0.037
Adjusted R^2	0.430	0.019	0.281	0.014	0.108	0.017
Residual Std. Error	0.001	0.001	0.002	0.001	0.002	0.002
F Statistic	7.786**	1.172	10.360***	1.339	6.960**	1.864

Note: *p<0.1; **p<0.05; ***p<0.01

This table reports the results of the cross-sectional regression tests for the Capital Asset Pricing Model (CAPM). At the end of each month, we sort the stocks into 10, 25 and 50 portfolios based on their model-implied beta (β^{fwd}), as well as the historical CAPM beta (β^{hist}) calculated using the past 252 trading days' returns. For each portfolio, we calculate the value-weighted portfolio beta measures and the value-weighted portfolio realized excess returns over the next month. We then regress the time-series mean of the portfolio excess returns on a constant and the time-series mean of portfolio beta measures. The sample period is from January 2004 to August 2023 and we use the stocks included in the S&P 500 index at each month-end. The standard errors are provided in parentheses. ***, **, and * denote statistical significance at the 1%, 5%, and 10% levels, respectively.

Figures

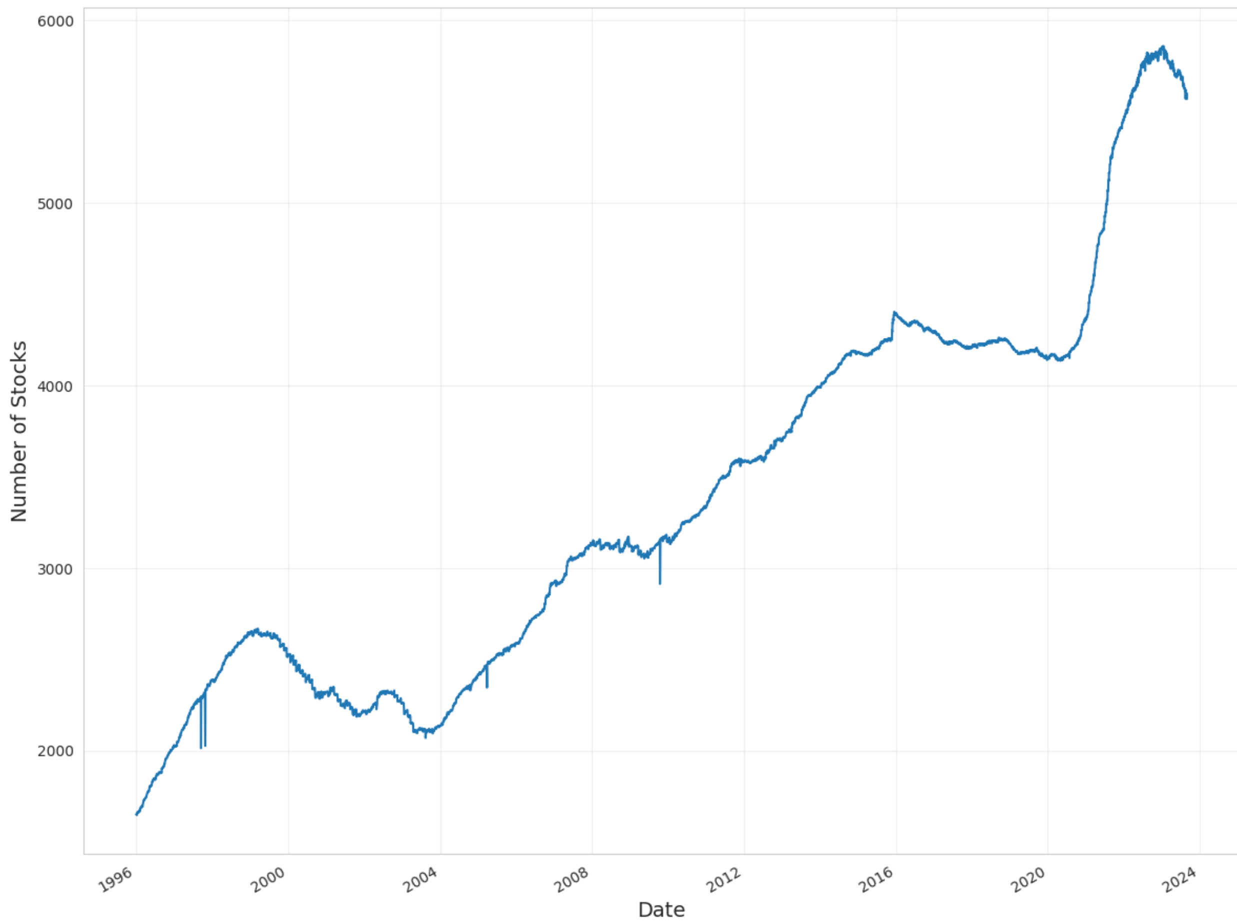

Figure 1: Number of Stocks with Option-Implied Volatility Surface Data (1996-2023). The figure shows the number of stocks with available option-implied volatility surface data from IvDB OptionMetrics, spanning from January 1996 to August 2023.

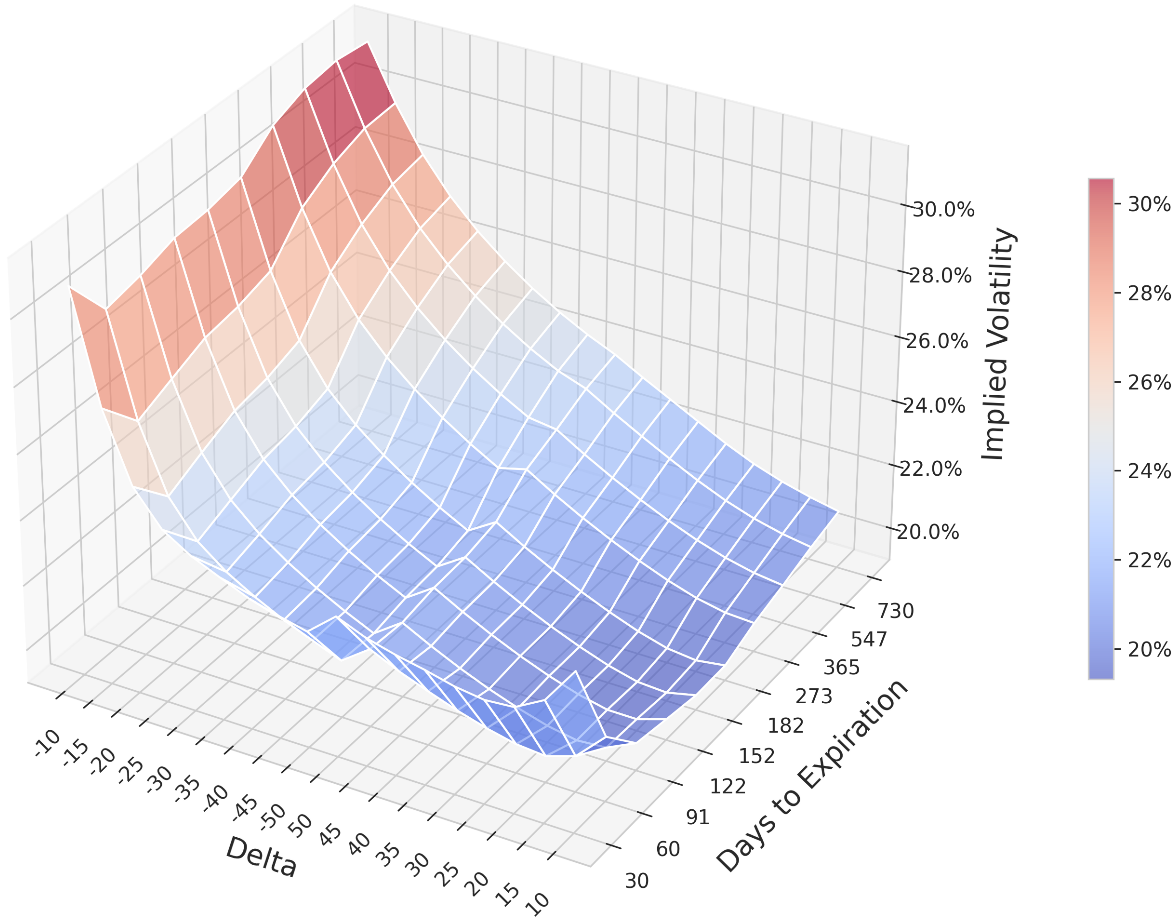

Figure 2: Example of Interpolated Option-Implied Volatility Surface (AAPL, 01/08/2023). The plot shows the volatility surface for Apple Inc. (AAPL) on Aug 1st, 2023, with moneyness (option delta) on the x-axis, days to expiration on the y-axis, and implied volatility levels represented by the color gradient.

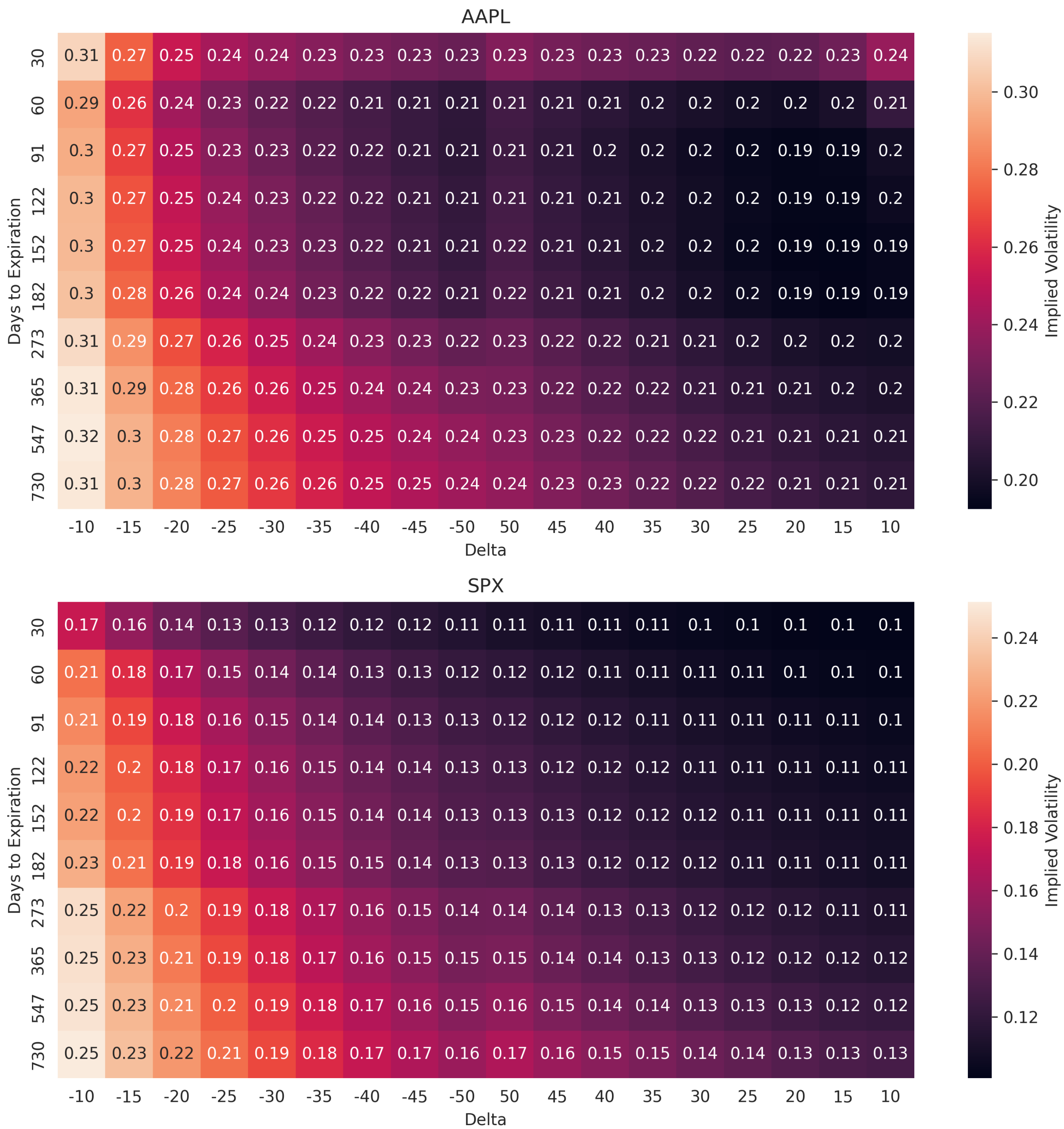

Figure 3: "Image" Representation of Volatility Surface (01/08/2023). The upper panel shows the heatmap of the volatility surface for Apple Inc. (AAPL), while the lower panel shows the volatility surface for the S&P 500 index (SPX) on the same date. The x-axis represents moneyness (option delta), and the y-axis represents days to expiration, with color intensity indicating implied volatility levels.

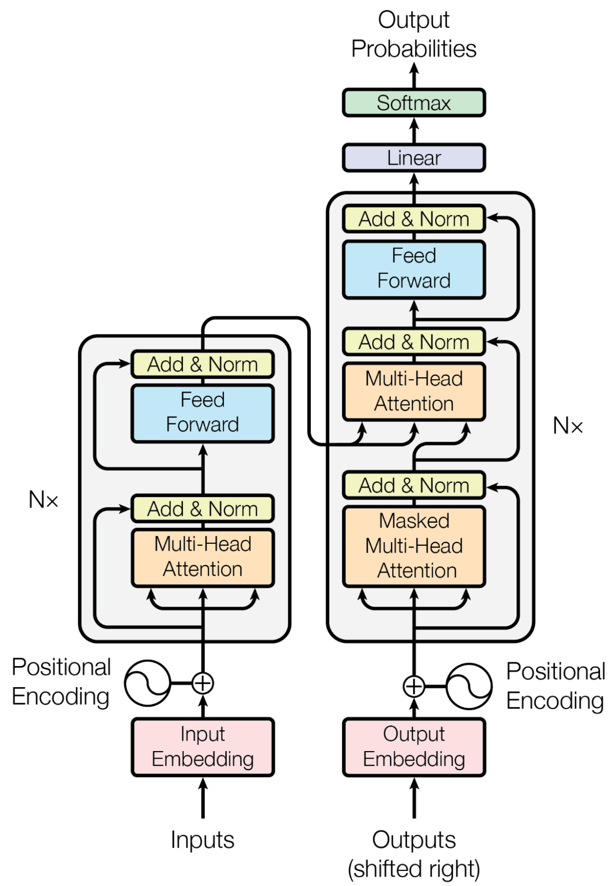

Figure 4: Transformer Architecture from Vaswani et al. (2017). The Transformer model consists of an encoder and a decoder, each composed of multiple layers of multi-head self-attention (MSA) and feed-forward neural networks (FFN). The encoder processes the input sequence, while the decoder generates the output sequence, attending to both the encoder's output and its own previous outputs.

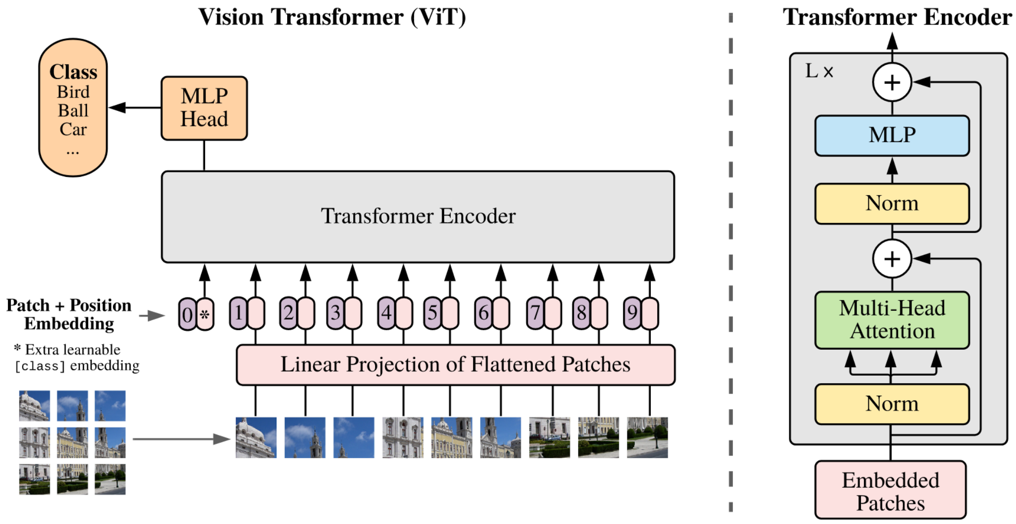

Figure 5: Vision Transformer (ViT) Architecture from Dosovitskiy et al. (2020). The image is split into fixed-size patches, linearly embedded, and then position embeddings are added. The resulting sequence of vectors is fed to a standard Transformer encoder. Similar to BERT, a classification token is prepended to the sequence, and the final hidden state corresponding to this token is used for classification tasks.

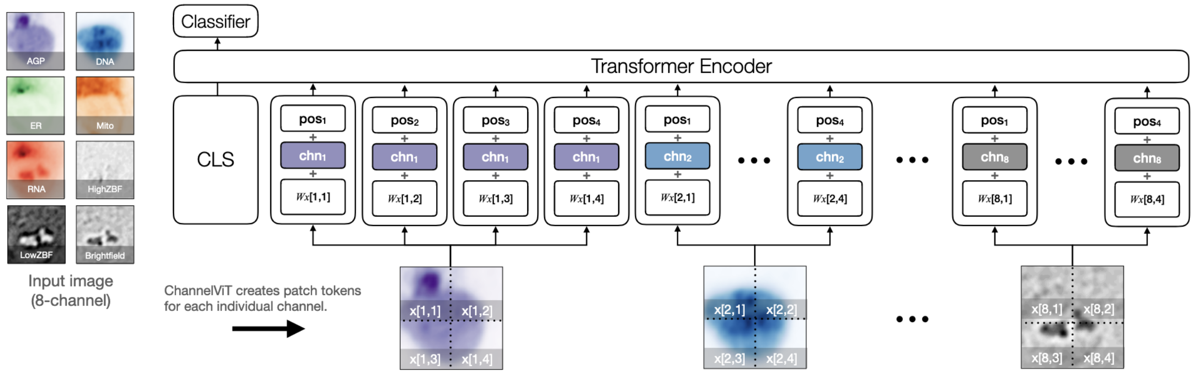

Figure 6: Channel Vision Transformer (ChannelViT) Architecture from Bao et al. (2024). The input image comprises multiple channels potentially carrying semantically distinct and independent information. ChannelViT constructs patch tokens for each individual channel, utilizing a learnable channel embedding **chn** to encode channel-specific information. The linear embedding **W** and positional embeddings **pos** are shared across all channels. The resulting sequence of vectors from all channels is concatenated and fed into a standard Transformer encoder. Similar to BERT, a classification token is prepended to the sequence, and the final hidden state corresponding to this token is used for classification tasks.

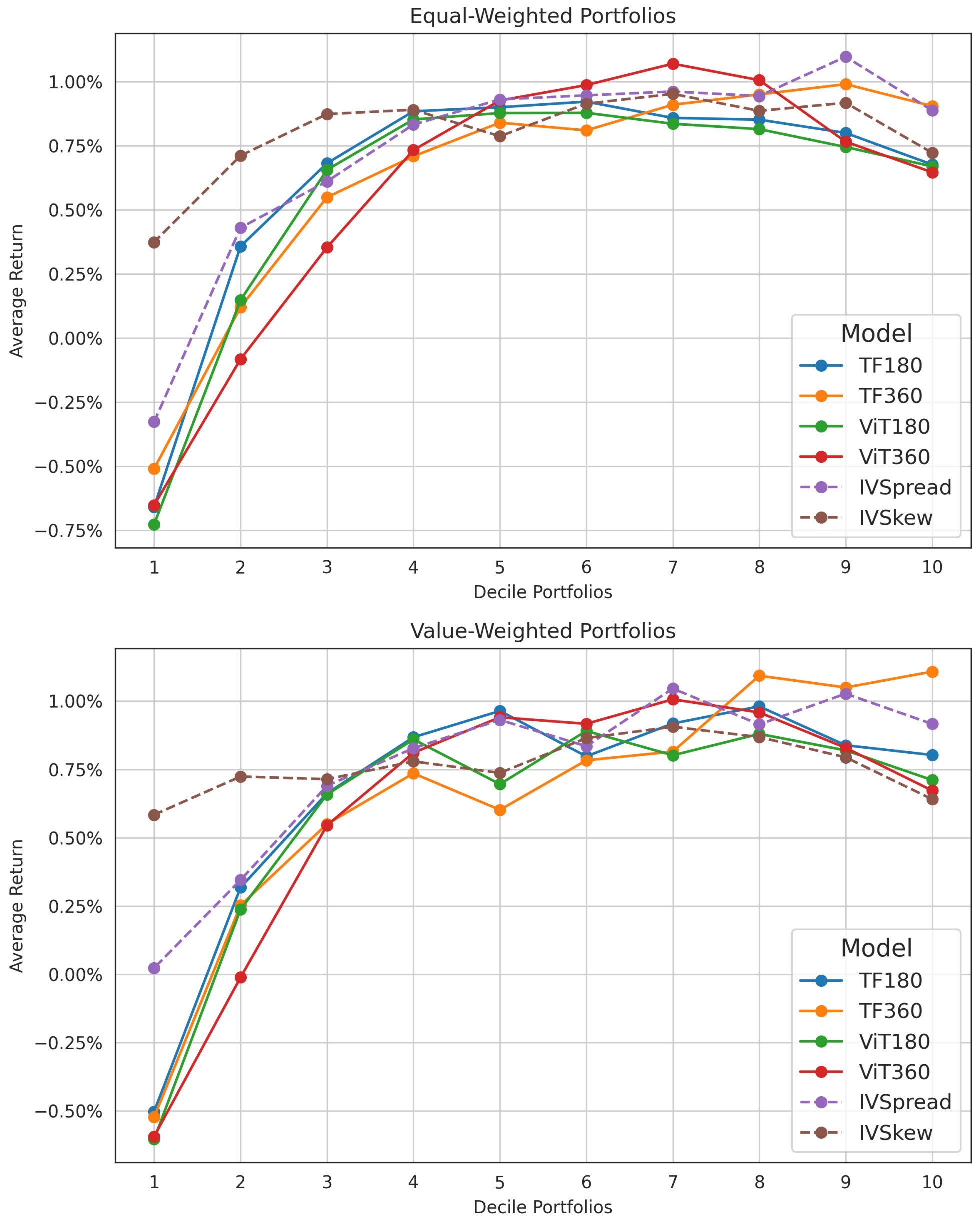

Figure 7: Portfolio Performance from Different Models. This figure shows the portfolio returns formed on out-of-sample predicted "up" probability $\mathbb{P}(r_{i,t}^{21} > 0)$ from different model specifications described in Section 4.1, as well as the **IVSpread** and **IVSkew** characteristics described in Appendix B. The upper panel shows the equal-weighted portfolio returns, while the lower panel shows the value-weighted portfolio returns. More details are provided in Table 1.

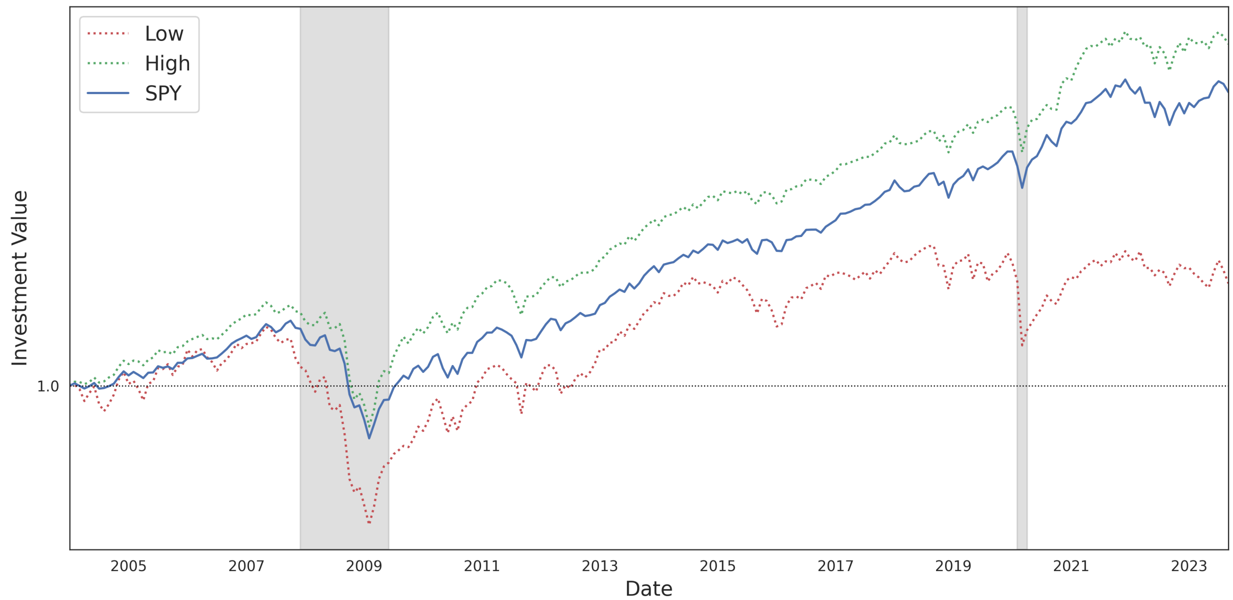

Figure 8: Cumulative Returns Performance. This figure shows the cumulative returns of equal-weight decile portfolios formed on out-of-sample predicted "up" probability $\mathbb{P}(r_{i,t}^{21} > 0)$ from January 2004 to August 2023 using the **ViT360** model described in Section 4.1. The sample includes only stocks that are part of the S&P 500 index. The "Low" series represents the 1/3 stocks with the lowest predicted "up" probabilities, while the "High" series represents the 2/3 stocks with the highest predicted "up" probabilities. The "SPY" series represents the cumulative return of the S&P 500 index ETF (SPY) over the same period.

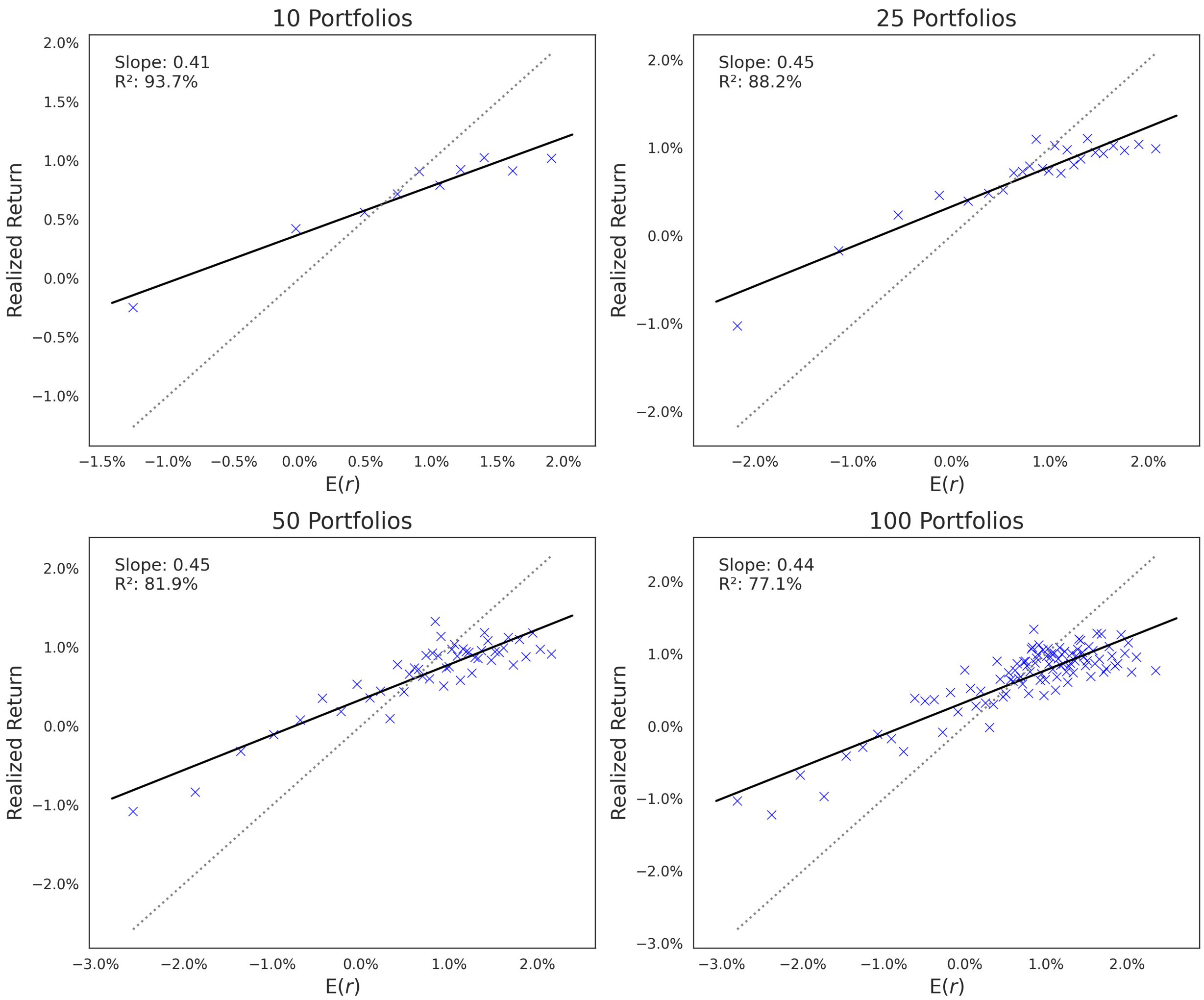

Figure 9: Portfolio sorted by expected stock returns $\mathbb{E}(r_{i,t}^{21})$. This figure shows the portfolio returns formed on out-of-sample predicted mean return $\mathbb{E}(r_{i,t}^{21})$ using the **ViT360** model with 100 quantile classes described in Section 4.2. The horizon is one month (21 trading days). At the end of each month, stocks are sorted into 10, 25, 50, or 100 portfolios based on their predicted mean returns $\mathbb{E}(r_{i,t}^{21})$. For each portfolio, we compute the value-weighted portfolio returns over the next month and the value-weighted portfolio average predicted mean return. We then plot the time-series average portfolio returns against the average predicted mean returns. The black line represents the fitted linear regression line, with the slope coefficient and R-squared value indicated in the legend. The grey dashed line represents the 45-degree line for reference. The sample period is from January 2004 to August 2023.

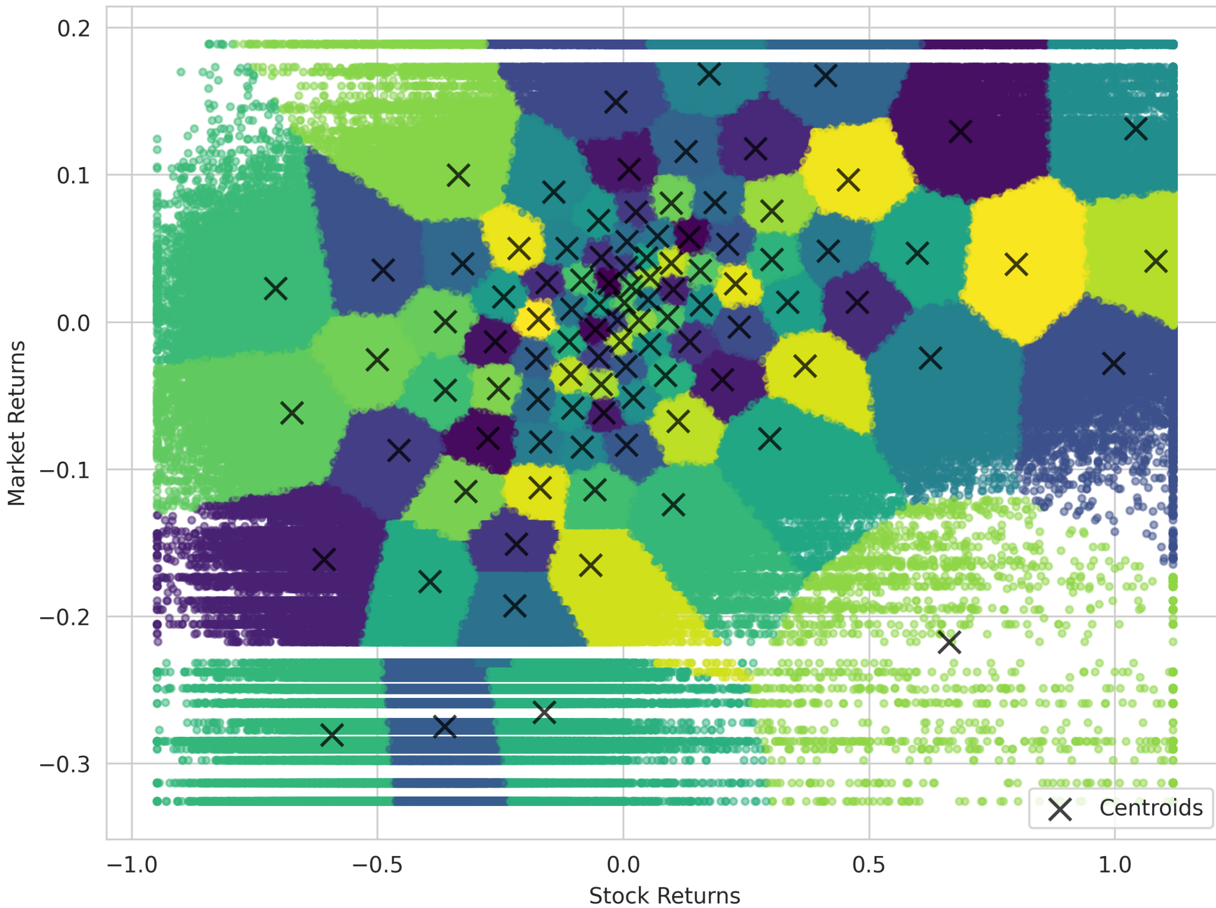

Figure 10: K-Means Clustering of Stock and Market Returns. This figure shows the K-Means clustering results with 100 clusters for the joint distribution of individual stock returns and market returns. Each point represents a joint observation of stock return and market return, colored according to its assigned cluster. The centroids of the clusters are marked with black crosses. The x-axis represents the individual stock returns, while the y-axis represents the market returns. The clustering is performed using the Euclidean distance in the two-dimensional return space. Both stock and market returns are winsorized at the 0.01% and 99.99% levels for extreme value control.

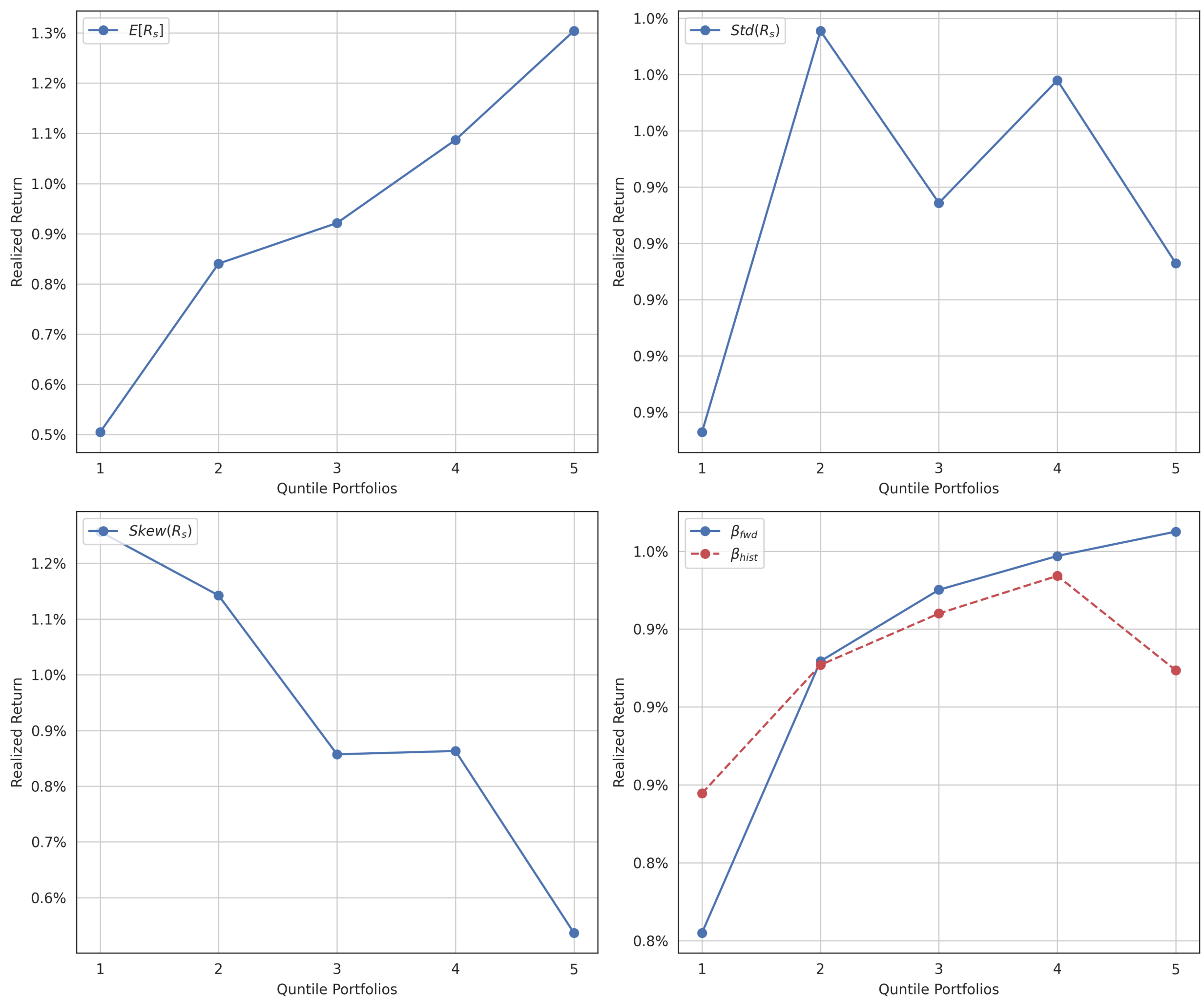

Figure 11: Portfolio Performance from K-Means Clustering (100 Clusters)

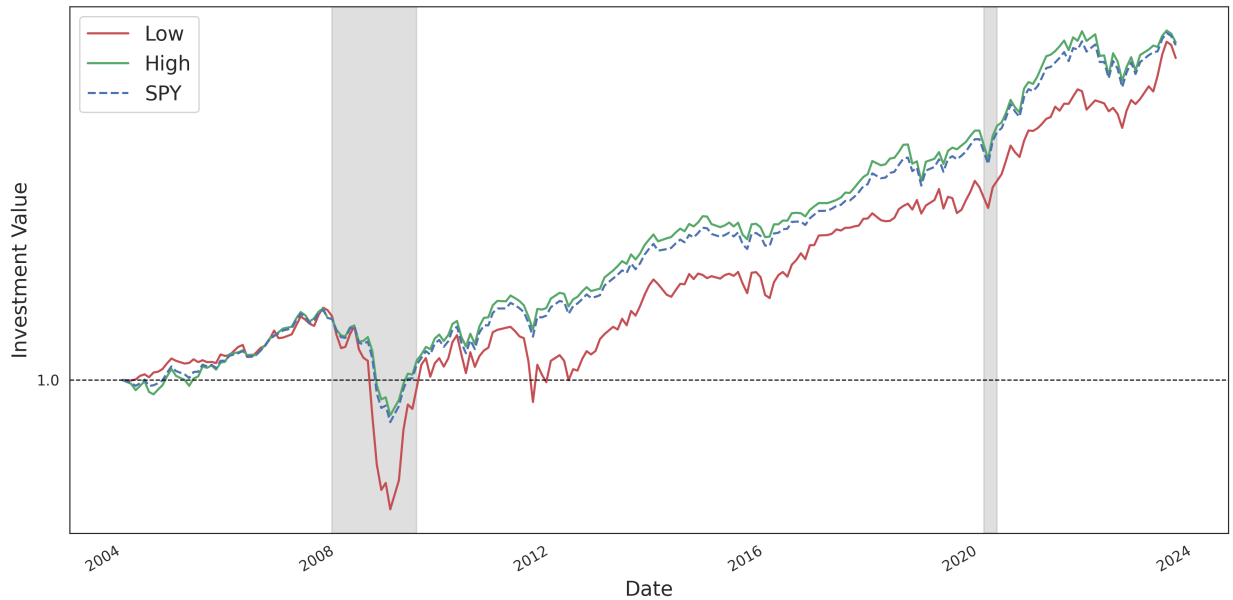

Figure 12: Cumulative Returns, S&P 500 Stocks. This figure shows the cumulative returns of equal-weight portfolios formed on out-of-sample predicted expected stock returns conditional on market crashes $\mathbb{E}(r_{i,t}^{21}|r_{m,t}^{21} \leq -0.2)$ using the **ViT360** model with K-Means clustering of 100 clusters described in Section 4.3. The sample includes only stocks that are part of the S&P 500 index. The "Low" series represents the 1/3 stocks with the lowest predicted conditional expected returns, while the "High" series represents the 2/3 stocks with the highest predicted conditional expected returns. The "SPY" series represents the cumulative return of the S&P 500 index ETF (SPY) over the same period.

A More about Transformer

A.1 Transformer Architecture

In this section, we provide a more detailed explanation of the Transformer architecture (Figure 4) introduced by Vaswani et al. (2017), and how we adapt it for our first model specification, the Temporal Transformer (TF) model.

- **Embeddings**: The input tokens are first converted to vectors of fixed dimension d_{model} through an learned embeddings. Denote the input sequence as $[\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^N]$, where each token $\mathbf{x}^i \in \mathbb{R}^{d_{input}}$. The embedding layer maps each token to a d_{model}-dimensional vector: $[\mathbf{x}^i] \rightarrow [\mathbf{x}^i\mathbf{E}]$, where $\mathbf{E} \in \mathbb{R}^{d_{input} \times d_{model}}$ is the learned embedding matrix. The resulting sequence of embeddings is $[\mathbf{x}^1\mathbf{E}; \mathbf{x}^2\mathbf{E}; ...; \mathbf{x}^N\mathbf{E}] \in \mathbb{R}^{N \times d_{model}}$.

- **Positional Encoding**: Since the Transformer architecture contains no recurrence or convolution, it lacks the inherent ability to capture the order of the input sequence. To address this, positional encodings are added to the input embeddings to provide information about the position of each token in the sequence. The positional encoding can be either learned or fixed, and Vaswani et al. (2017) proposed a fixed sinusoidal positional encoding defined as:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{24}$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{25}$$

  where pos is the position index, and i is the dimension index. The positional encodings are added to the input embeddings to form the final input to the encoder:

$$\mathbf{z}_0 = [\mathbf{x}^1\mathbf{E}; \mathbf{x}^2\mathbf{E}; ...; \mathbf{x}^N\mathbf{E}] + \mathbf{E}_{pos} \tag{26}$$

  where $\mathbf{E}_{pos} \in \mathbb{R}^{N \times d_{model}}$ is the positional encoding matrix.

- **Encoder**: The Transformer encoder consists of L identical layers, each containing two main sub-layers: Multi-Head Self-Attention (MSA) and a Position-wise Feed-Forward Networks (FFN). Each sub-layer is followed by a residual connection and layer normalization. The overall structure of the encoder can be summarized as follows:

  - **Multi-Head Self-Attention (MSA)**: This mechanism allows the model to focus on different parts of the input sequence simultaneously, capturing various aspects of the image. Each attention head computes a weighted sum of the input embeddings, where the weights are determined by the similarity between the query and key vectors. These similarity scores are calculated using Query (Q), Key (K), and Value (V) matrices, which are linear projections of the input embeddings. The attention scores are computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V \tag{27}$$

where d_k is the dimension of the key vectors, used for scaling. The multi-head attention mechanism allows the model to jointly attend to information from different representation subspaces at different positions. For layer $l = 1, ..., L$, the MSA block can be expressed as:

$$\mathbf{z}'_l = \text{LayerNorm}(\text{MSA}(\mathbf{z}_{l-1}) + \mathbf{z}_{l-1}) \tag{28}$$

where $\text{LayerNorm}(\cdot)$ denotes layer normalization, and $\mathbf{z}'_l \in \mathbb{R}^{N \times d_{model}}$ is the output of the MSA block at layer l.

– **Position-wise Feed-Forward Networks (FFN)**: Following the MSA, a fully connected feed-forward network is applied to each position independently and identically. This FFN typically consists of two linear layers with a non-linear activation function (e.g., ReLU) in between. The FFN can be expressed as:

$$\text{FFN}(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2 \tag{29}$$

$$\text{ReLU}(x) = \max(0, x) \tag{30}$$

where W_1, W_2 are weight matrices and b_1, b_2 are bias vectors. Besides, dropout is applied after each fully connected layer to prevent overfitting. For layer $l = 1, ..., L$, the FFN block can be expressed as:

$$\mathbf{z}_l = \text{LayerNorm}(\text{FFN}(\mathbf{z}'_l) + \mathbf{z}'_l) \tag{31}$$

where $\mathbf{z}_l \in \mathbb{R}^{N \times d_{model}}$ is the output of the MLP block at layer l.

- **Classification Head**: After passing through the L layers of the Transformer encoder, the final output $\mathbf{z}_L \in \mathbb{R}^{N \times d_{model}}$ is obtained. Since our model take the sequence as a time-series data, we select the last time step's output $\mathbf{z}_L^N \in \mathbb{R}^{d_{model}}$ as the encoded vector representation of the input sequence, assuming that it contains the most relevant information for prediction. This vector is then passed through a linear layer followed by a softmax function to produce the class probabilities:

$$\mathbf{p} = \text{softmax}(\mathbf{z}_L^N W_c + b_c) \tag{32}$$

$$\text{softmax}(\mathbf{a})_i = \frac{e^{a_i}}{\sum_{j=1}^{K} e^{a_j}} \tag{33}$$

where $W_c \in \mathbb{R}^{d_{model} \times K}$ and $b_c \in \mathbb{R}^K$ are the weights and bias of the linear layer, and K is the number of classes. The output $\mathbf{p} \in \mathbb{R}^K$ represents the predicted probabilities for each class.

A.2 Vision Transformer Architecture

The architecture of the ViT, shown in Figure 5, adapts the original Transformer in the following way:

- **Image Patching and Embedding**: The ViT model first reshapes the input image from a 2D grid of pixels into a sequence of flattened 2D patches. A image is typically represented as $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$, where (H, W) is the height and width of the original image, C is the number of channels (e.g., RGB). In the first stage, it is divided into N patches $[\mathbf{x}_p^1, \mathbf{x}_p^2, ..., \mathbf{x}_p^N]$, where each patch $\mathbf{x}_p^i \in \mathbb{R}^{P \times P \times C}$ is a small square region of the image with size (P, P) pixels, and $N = (H \times W)/(P^2)$ is the number of patches. These patches are then flattened and mapped to a latent D-dimensional embedding space through a trainable linear projection: $[\mathbf{x}_p^i] \rightarrow [\mathbf{x}_p^i E]$, where $E \in \mathbb{R}^{(P^2 \cdot C) \times d_{model}}$. This process converts the 2D spatial structure of the image into a 1D sequence of patch embeddings $[\mathbf{x}_p^1 E; \mathbf{x}_p^2 E; ...; \mathbf{x}_p^N E] \in \mathbb{R}^{N \times d_{model}}$, suitable for input into the Transformer encoder.

- **Learnable Class Token and Positional Encoding**: Inspired by the [CLS] token used in BERT (Bidirectional Encoder Representations from Transformers) model (Devlin et al., 2019), a learnable embedding is prepended to the sequence of patch embeddings. The state of this token at the output of the Transformer encoder serves as the aggregate image representation for classification tasks.

  Different from the original Transformer, where the sequence has a natural order (e.g., words in a sentence, or time steps in a time series), the sequence in ViT model represents image patches, which do not have an inherent order. We follow Dosovitskiy et al. (2020) and add learnable positional embeddings to the patch embeddings to retain spatial information about the position of each patch in the original image. This is crucial for the model to understand the spatial relationships between different patches.

The flattened patches, along with the class token and positional embeddings, are served as input to the Transformer encoder:

$$\mathbf{z}_0 = [\mathbf{x}_{class}; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; ...; \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos} \tag{34}$$

where $\mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times d_{model}}$ is the patch embedding projection matrix, $\mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times d_{model}}$ is the positional embedding matrix, and $\mathbf{z}_0 \in \mathbb{R}^{(N+1) \times d_{model}}$ is the resulting sequence of embedded patches, with length $N + 1$ (including the class token) and embedding dimension d_{model}.

- **Encoder**: The Encoder layers used in ViT are similar to the original Transformer model, but with some slight modifications:

$$\mathbf{z}'_l = \text{MSA}(\text{LayerNorm}(\mathbf{z}_{l-1})) + \mathbf{z}_{l-1} \tag{35}$$

$$\mathbf{z}_l = \text{FFN}(\text{LayerNorm}(\mathbf{z}'_l)) + \mathbf{z}'_l \tag{36}$$

In contrast to the post-normalization used in the original Transformer, the encoder blocks in ViT implement pre-normalization, where layer normalization is applied right before the MSA and FFN blocks, which has been shown to lead to more effective training for deeper networks (Xiong et al., 2020) without the need for learning rate warm-up. Besides, in the FFN block, GELU (Gaussian Error Linear Unit) activation function (Hendrycks and Gimpel, 2023) is used instead of ReLU.

- **Classification Head**: For image classification, only the output vector corresponding to the prepended class token in the final layer ($\mathbf{z}_L^0$) is used, which serves as the aggregate representation of the entire image:

$$\mathbf{p} = \text{softmax}(\text{LayerNorm}(\mathbf{z}_L^0)W_c + b_c) \tag{37}$$

Additional layer normalization is applied to the class token output before passing it through the linear classification head, because of the pre-normalization structure of the encoder.

B Option-Based and Other Characteristics

This section provides additional details on the option-based characteristics used in the empirical tests. Neuhierl et al. (2022) and Muravyev et al. (2025) summarized a comprehensive list of option-based characteristics that have been shown to be useful in explaining the cross-section of stock returns. We select a subset of most significant and widely used option characteristics. Most of these variables are calculated from the standardized option-implied volatility surface data (vsurfdYYYY), while the option volume are from the trading volume data (opvoldYYYY) and the realized volatility are from the historical volatility data (hvoldYYY). Monthly sample period is from January 1996 to August 2023, same as the date range of OptionMetrics data.

Thanks to the work of Chen and Zimmermann (2022) (CZ in the following), we are able to directly download most of these option characteristics from their website [8]. The methodology that they used closely followed the original papers with only minor modifications.

- **CIV** & **PIV**: Near At-The-Money (ATM) call and put implied volatility with maturity of 30 days and absolute delta of 0.5, following An et al. (2014).

- **ΔCIV** & **ΔPIV**: Monthly change in the near ATM call and put implied volatility, also from An et al. (2014).

- **SKEW**: Difference between average put and call IV, using options with 30 days to expiration and average across call with delta between 0.2 to 0.4, and put with delta between -0.2 to -0.4, following Neuhierl et al. (2022).

- **log(O/S)**: Total monthly option volume over all call and put options, divided by monthly stock trading volume, following Johnson and So (2012).

- **IVSpread**: Difference between the near ATM put (delta = -0.5 and maturity = 30 days) and call (delta = 0.5 and maturity = 30 days) implied volatility, using the last daily observation of each month, following Yan (2011) and Bali and Hovakimian (2009).

- **IVSkew**: Difference between the Out-The-Money (OTM) put implied volatility with moneyness closest to but above 1, and the OTM call implied volatility with moneyness closest to but below 1, using options with expiration between 10 and 60 days, following Xing et al. (2010).

- **VOV**: Volatility of option-implied volatility, calculated as the normalized standard deviation of daily implied volatility of near ATM options (delta = 0.5 for calls and

[8] `https://www.openassetpricing.com/`

delta = -0.5 for puts) with maturity of 30 days (Baltussen et al., 2018). Each month, the mean and standard deviation of daily IV are calculated, and VOV is defined as the standard deviation divided by the mean.

$$\text{VOV}_t = 0.5 * \left(\frac{\sigma(IV_{call,t})}{\mu(IV_{call,t})} + \frac{\sigma(IV_{put,t})}{\mu(IV_{put,t})} \right)$$

- **IVol-RVol**: Difference between option-implied volatility and realized volatility, following Bali and Hovakimian (2009). Realized volatility is calculated using daily stock returns over the past 30 calendar days, and implied volatility is the near ATM IV with maturity of 30 days, average across call and put options.

Besides these option-based characteristics, we also include the following commonly used stock characteristics as control variables in the empirical tests:

- **log(ME)**: Logarithm of market capitalization, the product of stock price and number of shares outstanding.
- **log(B/M)**: Logarithm of book to market ratio following Fama and French (1992).
- **Inv**: Annual growth rate of total asset following Cooper et al. (2008).
- **Profit**: Operating profitability, adjusted for R&D expenses, following Ball et al. (2016).
- **Illiq**: Amihud (2002) illiquidity measure, calculated as the past twelve-month average of absolute daily stock return divided by daily dollar trading volume.
- **Ivol**: Idiosyncratic volatility following Ang et al. (2006), calculated as the standard deviation of residuals from Fama-French three factor model over the past month daily data.
- β: CAPM beta calculated using the historical daily returns over a 252-trading-day estimation window and a 126-trading-day minimum requirement, provided by the Beta Suite by WRDS[9].
- $ret_{1,0}$: Short-term reversal, calculated as the stock return over the previous month (Jegadeesh, 1990).
- $ret_{12,2}$: Momentum, calculated as the stock return over the past 11 months, skipping the most recent month ($t-2$ to $t-12$), following Jegadeesh and Titman (1993).

[9] `https://wrds-www.wharton.upenn.edu/pages/get-data/beta-suite-wrds/`

C Additional Results

Table C.1: Portfolio Analysis, Binary Classification Model, Non-Micro Stocks

	Decile Portfolios										
Model	1	2	3	4	5	6	7	8	9	10	10 - 1
					Equal-Weighted Portfolios						
TF180	0.12	0.69	0.86*	0.95**	0.95**	0.87**	0.91***	0.85***	0.80***	0.72***	0.59
	(0.22)	(1.33)	(1.90)	(2.33)	(2.50)	(2.52)	(2.82)	(2.90)	(3.18)	(3.49)	(1.42)
TF360	0.03	0.60	0.71	0.87**	0.80**	0.90***	0.92***	0.93***	0.99***	0.96***	0.93**
	(0.05)	(1.27)	(1.58)	(2.19)	(2.18)	(2.64)	(2.82)	(3.11)	(3.32)	(2.99)	(2.34)
ViT180	0.09	0.70	0.82*	0.89**	0.88**	0.82**	0.86***	0.83***	0.76***	0.69***	0.60
	(0.15)	(1.40)	(1.89)	(2.23)	(2.45)	(2.37)	(2.69)	(2.99)	(2.98)	(3.35)	(1.37)
ViT360	-0.19	0.49	0.65*	0.87**	0.89**	0.96***	1.03***	1.05***	0.86**	0.73**	0.92***
	(-0.40)	(1.12)	(1.72)	(2.35)	(2.52)	(2.64)	(2.81)	(3.30)	(2.43)	(2.13)	(2.65)
					Value-Weighted Portfolios						
TF180	0.32	0.61	0.83*	0.92**	0.82**	0.91***	0.91***	0.96***	0.77***	0.78***	0.46
	(0.52)	(1.10)	(1.75)	(2.14)	(2.03)	(2.69)	(2.71)	(3.25)	(3.28)	(4.12)	(0.92)
TF360	0.10	0.67	0.54	0.64	0.70*	0.79***	0.98***	0.99***	0.98***	1.05***	0.94*
	(0.17)	(1.26)	(1.09)	(1.49)	(1.82)	(2.64)	(2.90)	(3.46)	(3.76)	(3.46)	(1.90)
ViT180	0.04	0.86	0.79	0.85*	0.64*	0.87***	0.79**	0.91***	0.78***	0.70***	0.66
	(0.06)	(1.48)	(1.63)	(1.82)	(1.72)	(2.75)	(2.36)	(3.48)	(3.03)	(3.48)	(1.25)
ViT360	-0.13	0.68	0.81*	0.84**	0.90***	0.89**	0.98***	1.05***	0.79**	0.73*	0.86**
	(-0.25)	(1.58)	(1.86)	(2.31)	(2.59)	(2.48)	(2.74)	(3.24)	(2.41)	(1.87)	(2.38)

Note: $^{***}p < 0.01$; $^{**}p < 0.05$; $^{*}p < 0.1$

Table C.2: Time Series Regression, Binary Classification Model, Non-Micro Stocks

	TF180	TF360	ViT180	ViT360	IVSpread	IVSkew
	(1)	(2)	(3)	(4)	(5)	(6)
Alpha	0.006**	0.011***	0.008***	0.009***	0.006***	-0.000
	(0.003)	(0.003)	(0.003)	(0.003)	(0.001)	(0.002)
MKT	-0.630***	-0.708***	-0.738***	-0.339***	-0.035	0.037
	(0.100)	(0.139)	(0.063)	(0.104)	(0.045)	(0.054)
SMB	-0.626***	-0.639***	-1.038***	-0.862***	0.024	-0.126
	(0.115)	(0.163)	(0.128)	(0.154)	(0.078)	(0.118)
HML	0.461***	0.358*	0.390**	0.426*	0.093	-0.290***
	(0.124)	(0.204)	(0.155)	(0.248)	(0.110)	(0.092)
RMW	0.953***	0.880***	1.117***	0.711***	0.343***	-0.032
	(0.203)	(0.237)	(0.219)	(0.263)	(0.111)	(0.154)
CMA	0.082	0.005	0.325	-0.359	-0.056	-0.278*
	(0.213)	(0.330)	(0.282)	(0.379)	(0.182)	(0.142)
UMD	0.130	-0.187	0.223	-0.156	0.071*	0.150*
	(0.180)	(0.204)	(0.178)	(0.186)	(0.038)	(0.084)
Observations	220	220	236	236	228	228
R^2	0.591	0.468	0.677	0.270	0.091	0.235
Adjusted R^2	0.580	0.453	0.668	0.251	0.067	0.215
Residual Std. Error	0.041	0.050	0.044	0.059	0.024	0.030
F Statistic	31.930***	23.746***	106.589***	9.118***	3.055***	8.704***

Note: *p<0.1; **p<0.05; ***p<0.01

Table C.3: Fama-MacBeth Regression, Binary Classification Model, Non-Micro Stocks

	TF180		TF360		ViT180		ViT360	
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
$\hat{P}(r_{i,t}^{21} > 0)$	0.034	-0.004	0.067	0.070	0.048	0.045	0.064***	0.057**
	(0.036)	(0.028)	(0.073)	(0.055)	(0.042)	(0.033)	(0.024)	(0.025)
CIV		-0.033***		-0.028**		-0.027***		-0.022**
		(0.011)		(0.011)		(0.010)		(0.011)
PIV		0.020*		0.018*		0.021*		0.021*
		(0.012)		(0.011)		(0.012)		(0.013)
SKEW		-0.006		-0.005		-0.007		-0.004
		(0.009)		(0.009)		(0.008)		(0.008)
ΔCIV		-0.009		-0.010		-0.010		-0.009
		(0.008)		(0.008)		(0.008)		(0.008)
ΔPIV		0.023**		0.021**		0.020**		0.017*
		(0.010)		(0.010)		(0.009)		(0.009)
log(O/S)		0.000		-0.000		0.000		0.000
		(0.000)		(0.000)		(0.000)		(0.000)
IVSpread		-0.062***		-0.060***		-0.061***		-0.057***
		(0.014)		(0.013)		(0.013)		(0.012)
IVSkew		-0.006		-0.006		-0.008		-0.005
		(0.007)		(0.007)		(0.006)		(0.006)
IVol-RVol		0.005		0.005		0.004		0.004
		(0.004)		(0.004)		(0.004)		(0.004)
beta		0.001		0.002		0.001		0.001
		(0.003)		(0.003)		(0.003)		(0.003)
log(ME)		-0.000		-0.001		-0.001		-0.000
		(0.001)		(0.001)		(0.001)		(0.000)
N	539,932	288,990	539,932	288,990	581,697	309,011	581,697	309,011
Adj R^2	0.027	0.073	0.027	0.074	0.034	0.071	0.028	0.071

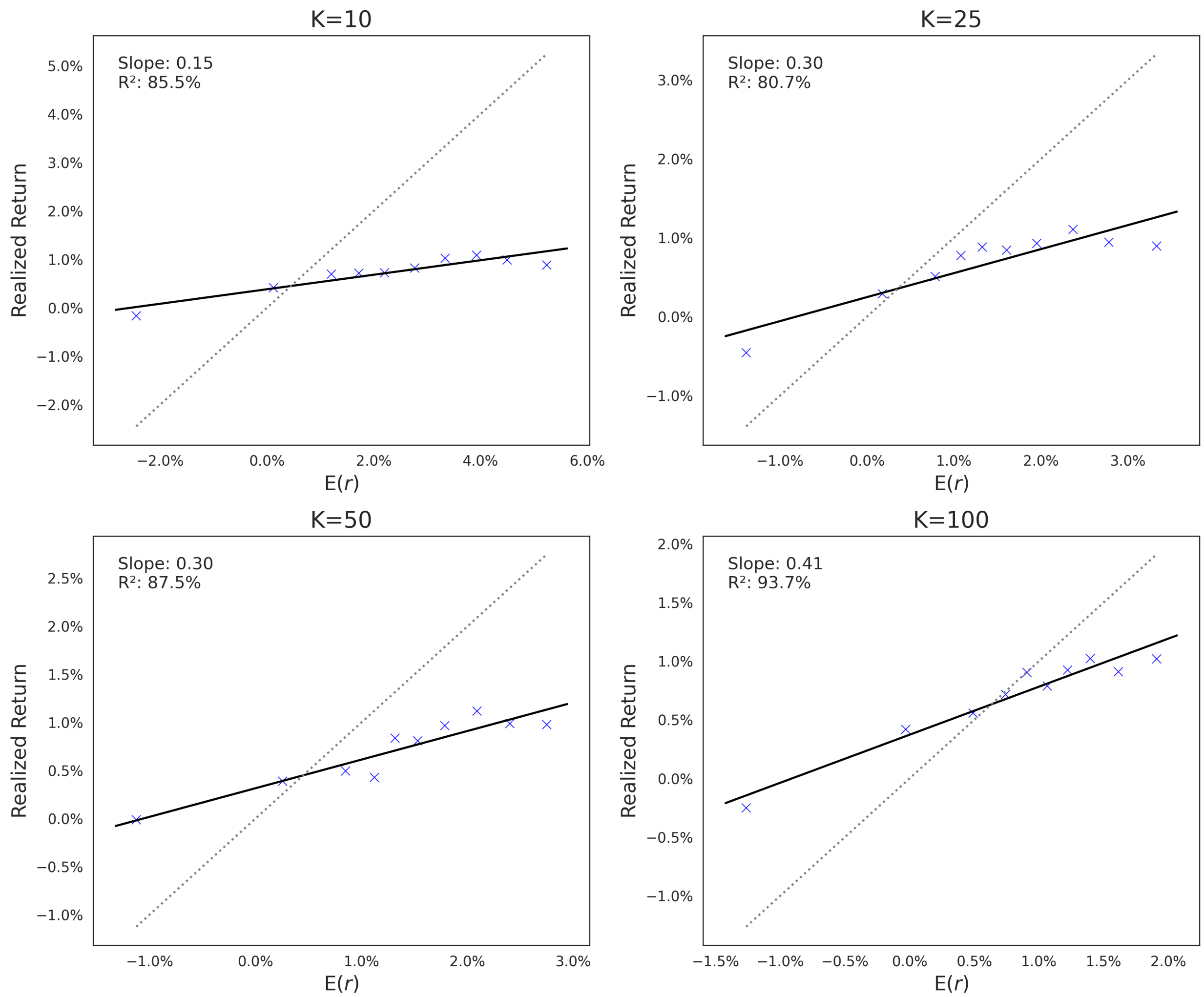

Figure C.1: Portfolio sorted by expected stock returns, for different number of classes (K).

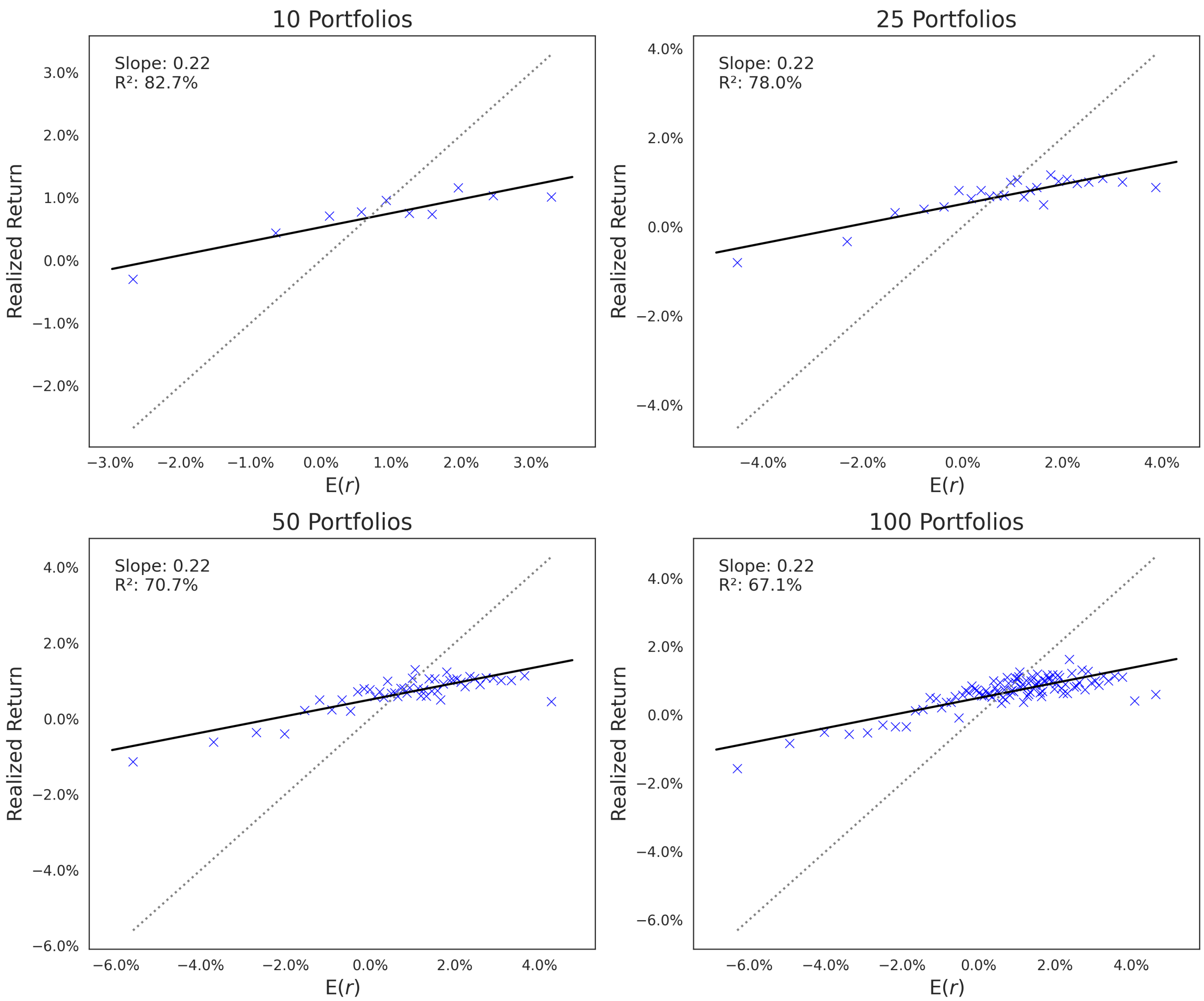

Figure C.2: Portfolio sorted by expected stock returns, **TF360** model.

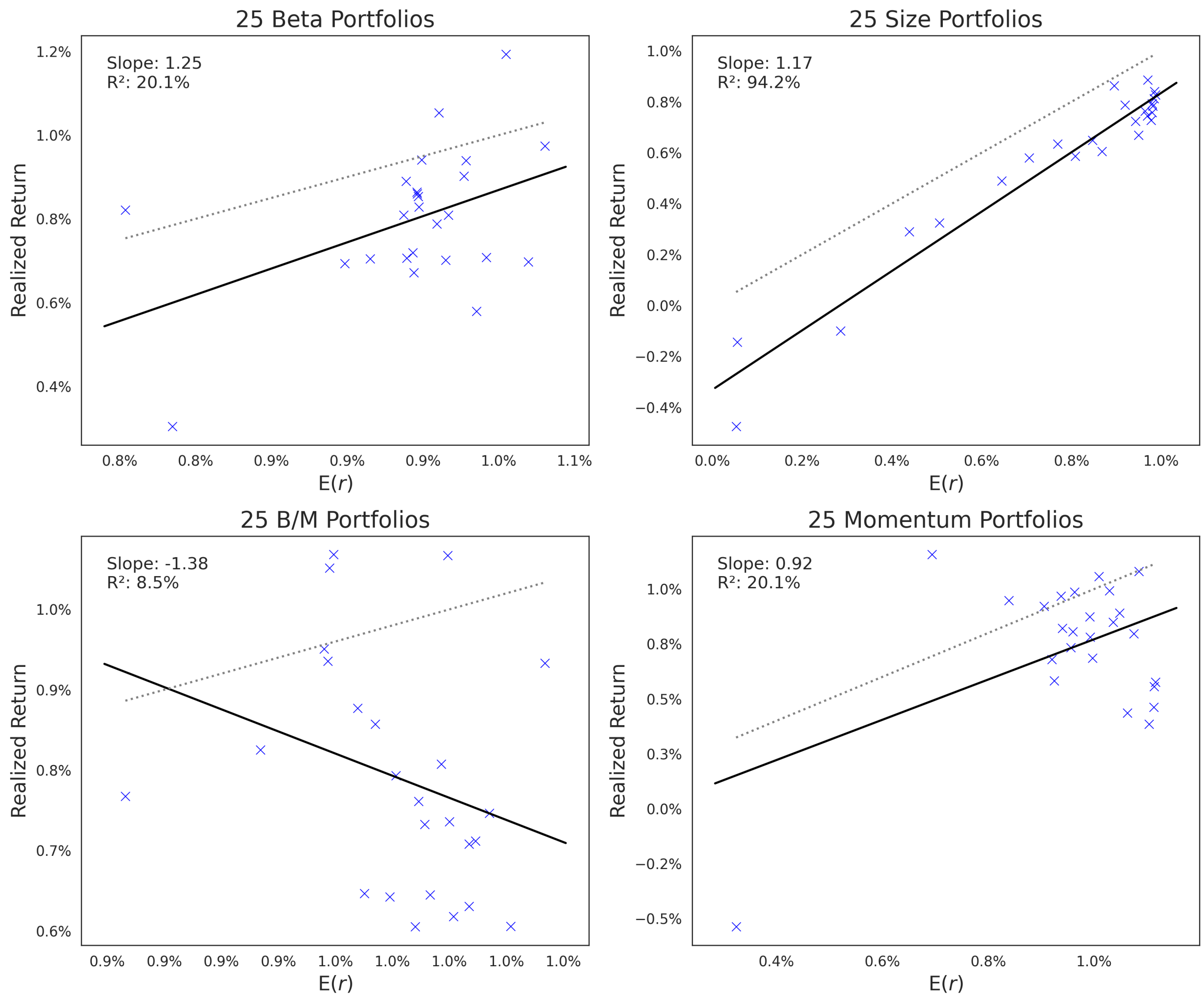

Figure C.3: Portfolio sorted by beta, size, book-to-market, momentum.