Learning Atmospheric Boundary Layer Turbulence

Sara Shamekh[1] and Pierre Gentine[2]

[1]Ecole Normale Superieure
[2]Columbia University

June 23, 2023

Abstract

Accurately representing vertical turbulent fluxes in the planetary boundary layer is vital for moisture and energy transport. Nonetheless, the parameterization of the boundary layer remains a major source of inaccuracy in climate models. Recently, machine learning techniques have gained popularity for representing oceanic and atmospheric processes, yet their high dimensionality limits interpretability. This study introduces a new neural network architecture employing non-linear dimensionality reduction to predict vertical turbulent fluxes in a dry convective boundary layer. Our method utilizes turbulent kinetic energy and scalar profiles as input to extract a physically constrained two-dimensional latent space, providing the necessary yet minimal information for accurate flux prediction.

We obtained data by coarse-graining Large Eddy Simulations covering a broad spectrum of boundary layer conditions, from weakly to strongly unstable. These regimes are employed to constrain the latent space disentanglement, enhancing interpretability. By applying this constraint, we decompose the vertical turbulent flux of various scalars into two main modes of variability: wind shear and convective transport.

Our data-driven parameterization accurately predicts vertical turbulent fluxes (heat and passive scalars) across turbulent regimes, surpassing state-of-the-art schemes like the eddy-diffusivity mass flux scheme. By projecting each variability mode onto its associated scalar gradient, we estimate the diffusive flux and learn the eddy diffusivity. The diffusive flux is found to be significant only in the surface layer for both modes and becomes negligible in the mixed layer. The retrieved eddy diffusivity is considerably smaller than previous estimates used in conventional parameterizations, highlighting the predominant non-diffusive nature of transport.

Learning Atmospheric Boundary Layer Turbulence

S. Shamekh[1,2], P. Gentine[1,2]

[1]Department of Earth and Environmental Engineering, Columbia University, New York, NY, USA
[2]Center for Learning the Earth with Artificial Intelligence And Physics (LEAP), Columbia University, New York, NY, USA

Key Points:

- We propose a physics-informed machine learning technique to predict the vertical turbulent fluxes in the planetary boundary layer
- The vertical turbulent fluxes are decomposed into wind shear and convective modes and their contributions to flux generation are approximated
- The vertical turbulent fluxes exhibit a non-diffusive nature with the estimated eddy diffusivity significantly smaller than previous estimates

Corresponding author: Sara Shamekh, `ss6287@columbia.edu`

Abstract

Accurately representing vertical turbulent fluxes in the planetary boundary layer is vital for moisture and energy transport. Nonetheless, the parameterization of the boundary layer remains a major source of inaccuracy in climate models. Recently, machine learning techniques have gained popularity for representing oceanic and atmospheric processes, yet their high dimensionality often limits interpretability. This study introduces a new neural network architecture employing non-linear dimensionality reduction (encoder-decoder) to accurately predict vertical turbulent fluxes in a dry convective boundary layer. Our method utilizes the vertical profiles of turbulent kinetic energy and scalars as input to extract a physically constrained two-dimensional latent space, providing the necessary yet minimal information for accurate flux prediction. For this study, we obtained data by coarse-graining Large Eddy Simulations covering a broad spectrum of boundary layer conditions, ranging from weakly to strongly unstable. These regimes, driven by shear or buoyancy, are employed to constrain the latent space disentanglement, enhancing interpretability. By applying this constraint, we decompose the vertical turbulent flux of various scalars into two main modes of variability: one associated with wind shear and the other with convective transport. Our data-driven parameterization accurately predicts vertical turbulent fluxes (heat and passive scalars) across turbulent regimes, surpassing state-of-the-art schemes like the eddy-diffusivity mass flux scheme. By projecting each variability mode onto its associated scalar gradient, we estimate the diffusive flux and learn the eddy diffusivity. The diffusive flux is found to be significant only in the surface layer for both modes and becomes negligible in the mixed layer. The retrieved eddy diffusivity is considerably smaller than previous estimates used in conventional parameterizations, highlighting the predominant non-diffusive nature of transport.

Plain Language Summary

This study focuses on better understanding and predicting the movement of moisture and energy in the lower part of the Earth's atmosphere, called the planetary boundary layer. This is important as it directly impacts our ability to make accurate weather forecasts and model the climate. The study utilizes neural networks to analyze extensive data derived from computer simulations of the atmosphere. The objective is to extract meaningful insights from this complex data and facilitate accurate predictions. To achieve this, we employ an advanced form of neural networks, called encoder-decoder, that is a dimensionality reduction technique. This approach aims to distill the most crucial information from the data while maintaining simplicity and interpretability. Through this process, the neural network effectively reduces the data to two key factors influencing the movement of moisture and energy: wind shear (variations in wind speed and direction) and convective transport (movement resulting from heating and cooling). Overall, this study demonstrates that employing machine learning techniques can significantly advance our understanding and prediction of the intricate processes occurring in the atmosphere. This, in turn, leads to the development of more precise climate models and improved weather forecasts.

1 Introduction

In the planetary boundary layer (PBL), turbulence occurs over a wide range of scales, causing the mixing and transport of moisture, heat, momentum, and chemical scalars (Stull, 1988). An accurate representation of turbulent mixing is crucial for predicting many critical climate processes, such as low clouds, lower free tropospheric humidity and temperature, air-sea interaction, and more (Stensrud, 2009). Climate and weather models, which use a discretized spatiotemporal representation of the physical equations, cannot resolve scales smaller than their grid size. Therefore, these models rely on parameterization, an approximation of the impact of unresolved physical processes based on resolved quantities, such as turbulent mixing occurring at unresolved scales and transporting momentum, energy and scalars.

Traditionally, boundary layer turbulent mixing was first assumed to behave as a diffusion and therefore to be occurring down local gradient:

$$\overline{w'x'} = -K\frac{d\overline{X}}{dz} \tag{1}$$

Where $K(m^2s^{-1})$ is called the eddy diffusivity, w is the vertical velocity, and X represents a scalar variable that is being transported by the flow. Over-line indicates a horizontal averaging, and prime is the deviation from the spatial average: $x\prime = X - \overline{X}$.

Although simple and intuitive, this scheme fails to accurately predict the turbulent heat flux in the mixed layer of the convective boundary layer, where a zero or positive gradient of potential temperature coexists with finite and positive heat flux (Corrsin, 1975; Stull, 1988). This positive heat flux has been associated with the impact of large turbulent coherent structures, such as updrafts and downdrafts (Park et al., 2016), that are ubiquitous in the convective boundary layer and connect the surface layer to the top of the boundary layer by transporting heat and other variables upward, quickly within a model time step. Rising updrafts are accompanied by a descending counterpart in the convective boundary layer, and by a top-of-the-boundary layer entrainment flux occurring between the weakly turbulent stable stratification above the boundary layer and the convective layer (Fedorovich et al., 2004; Gentine et al., 2015). Large eddies traveling over large distances do not respect the eddy diffusion local gradient perspective, as these coherent structures bring non-locality to the turbulent fluxes.

Over the past few decades, several approaches have been proposed to correct the eddy-diffusion approach and include the effect of non-local eddies in turbulent flux parameterization, mainly considering the non-locality by adding a non-local term to the eddy diffusion (Ertel, 1942; Priestley & Swinbank, 1947). A few examples of such approaches are the eddy diffusivity – counter-gradient, hereafter EDCG, (J. Deardorff, 1972; Troen & Mahrt, 1986; Holtslag & Moeng, 1991), the transport asymmetry (Moeng & Wyngaard, 1984, 1989; Wyngaard & Brost, 1984; Wyngaard & Weil, 1991; Wyngaard & Moeng, 1992), or the eddy diffusivity – mass flux (Siebesma & Cuijpers, 1995; Siebesma & Teixeira, 2000; Siebesma et al., 2007), which is now widely used in weather and climate models. While a thorough review of the vertical turbulent parameterization is out of the scope of this work, we briefly discuss the eddy diffusivity – mass flux (EDMF, Siebesma et al. (2007)) approach since it is widely used and several EDMF versions have been developed and implemented in operational weather forecasts and climate models. Thus, we will use this as a benchmark to evaluate our parameterization for modeling vertical turbulent fluxes.

The EDMF model assumes that the total vertical flux of a scalar (e.g., heat, moisture) is due to the contribution of strongly convective updrafts, which cover a negligible horizontal fractional area, and a complementary slowly subsiding environment, with negligible vertical velocity. The total flux of scalar X can then be written as:

$$\overline{w'x'} = a_u\overline{w'x'}^u + (1-a_u)\overline{w'x'}^e + a_u(w_u - \overline{w})(X_u - X_e) \tag{2}$$

where u and e represent the updraft and environment, respectively. a_u is the updraft fractional area. w_u and $\overline{w}$ are the mean vertical velocity over the updraft and environment, and X_u and X_e are the corresponding mean scalar. Assuming a small fractional area coverage of the updrafts and a negligible vertical velocity in the environment, we can eliminate the first term on the RHS, approximate $\overline{w}$ to be zero, and replace X_e with $\overline{X}$. Thus Equation 2 reduces to:

$$\overline{w'x'} \approx \overline{w'x'}^e + a_u w_u(X_u - \overline{X}) \tag{3}$$

The first term on the RHS of Equation 3 is modeled using an eddy diffusivity (Equation 1) and the second term is the mass flux, non-local, contribution to total vertical turbulent flux, which was inspired by modeling of deep convection (Betts, 1973) .

Despite its successes in improving purely convective boundary layer parameterization compared to other approaches (e.g, pure ED or EDCG), EDMF still has important shortcomings. First, the EDMF decomposes the total flux into ED, modelling small scale eddies, and MF, modelling large scale updrafts. However, these two terms are not coupled in any systematic way, a theory for the relative partitioning between these two contributions does not exist, and a theory for an optimal scale at which the continuous spectrum of boundary layer eddies can be divided into small eddies and large thermals has not been established. Additionally, one of the main assumptions in deriving Equation 3 is that the updraft fractional area is negligible. However, recent studies (Q. Li et al., 2021; Chinita et al., 2018; Park et al., 2016) suggest a fractional area of 20-30 percent. Consequently, some of approximations made to derive the two-term Equation 3 does not hold accurately. For instance, the first term in the RHS of Equation 2 has been shown to be important and responsible for local fluxes in updrafts (Q. Li et al., 2021), or X_u may have a non-negligible impact on the domain mean value $\overline{X}$. Furthermore, the original EDMF schemes have been developed for a purely convective boundary layer (Siebesma et al., 2007; Soares et al., 2004), i.e., with small wind shear, thus EDMF poorly generalizes to situations driven by both wind and convection (Kalina et al., 2021). Some models, employ a hybrid scheme, such that, for weakly convective cases, they use EDCG and, at a certain instability threshold, they switch to EDMF (Han et al., 2016). However, this threshold is set arbitrarily and the switch between parameterizations appears quite ad hoc, and rather, a unified treatment of turbulence would be preferred.

In addition, one of the main pitfalls of the EDMF approach is its lack of explicit treatment of boundary layer top entrainment processes, which ventilate and mix air from the lower troposphere into the boundary layer. Entrainment significantly impacts the growth and structure of the PBL (Angevine et al., 1994), the evolution of mixed layer properties, surface fluxes, and the formation and maintenance of shallow clouds (Haghshenas & Mellado, 2019). However, EDMF does not explicitly take entrainment into account, which is potentially one reason for its shortcomings in accurately predicting turbulent fluxes at the top of the PBL and the exchange of PBL and lower troposphere. For instance, at the European Center for Medium Weather Forecast, entrainment is added (as a fraction of the surface buoyancy flux) as a diagnostic correction term to the EDMF model to obtain reasonable diurnal growth of the PBL. Additionally, wind shear strongly affects the entrainment flux and should be accounted for along with (dry) convection (Haghshenas & Mellado, 2019). Therefore, a more complete treatment of turbulence in the PBL is required, ideally one that can account for varying regimes from shear- to convectively-driven conditions and all forms of transport in the boundary layer, including eddies driven by shear or convection and entrainment at the top of the boundary layer.

Machine learning has proven to be a powerful tool for parameterizing subgrid-scale processes in the atmosphere and the ocean, particularly with the rise in popularity of neural networks (NNs) and deep learning as well as the explosion of high-resolution simulation data. In the field of atmosphere and ocean modeling, deep neural networks have shown significant potential in replacing traditional parameterizations of unresolved subgrid-scale processes (Gentine et al., 2018; Rasp et al., 2018; Mooers, Pritchard, et al., 2021; Yuval & O'Gorman, 2020; Bolton & Zanna, 2019; Shamekh et al., 2022; Perezhogin et al., 2023) due to their power in approximating a non-linear mapping between observed and unobserved quantities. Using ocean data, convolutional NNs have been shown to accurately predict subgrid-scale turbulent fluxes when trained on coarse-scale data (Bolton & Zanna, 2019), which could account for the spatial auto-correlation in the input data. In a similar vein, Cheng et al. (2019) used Direct Numerical Simulation (DNS) data of the planetary boundary layer to train a neural network that outperforms popular Large

Eddy Simulation (LES) schemes like the Smagorinsky (Smagorinsky, 1963) and Smagorinsky-Bardina (Bardina et al., 1980) turbulent flux models.

The work mentioned above showed promise in using neural networks in climate and weather models to replace traditional parameterization. One avenue that deserves more exploration is the use of interpretable machine learning models tailored to the problem of interest and including physical constraints, as they could unveil new understanding of the underlying physics. One such candidate could be a reduced order model (ROM) that relies on the fact that even high-dimensional complex flows often exhibit a few dominant modes of variability (Taira et al., 2017) that can provide coarse but key information about the flow. Encoder-decoder and variational auto-encoder (VAE) (Kingma & Welling, 2022) are powerful examples of ROM that map high-dimensional complex data to a low-dimensional latent representation. This latent representation captures the dominant modes of variability or structure in the data and because of its reduced dimension, can be much more interpretable. Mooers, Tuyls, et al. (2021) showed that VAEs could reconstruct velocity fields from a super-parameterized storm-resolving model. Additionally, they showed that the latent space could be categorized into different clusters, each representing a specific convection regime. Behrens et al. (2022) took this approach further and showed that VAE could reconstruct large-scale variables and map the latent variables to convection tendencies. They found that each latent variable represented a specific type or aspect of convection

In this work, we use encoder-decoder models and present a novel approach to data-driven parameterization of turbulence in the convective boundary layer, collapsing the complexity of turbulence into a few dimensions: the latent space. This latent space's dimensions are then disentangled using physical constraints based on the forcing of the boundary layer regimes: wind shear and surface heating. This constraint allows us to decompose the total flux of a scalar into two modes: one related to wind shear; the other related to convection. We use encoder-decoder models to approximate the latent representations of the scalars and Turbulent Kinetic Energy (TKE) profiles and then use these representations to predict the corresponding turbulent fluxes and modes of variability. Using this neural network, we aim to achieve the following objectives:

1. Predicting the vertical turbulent flux of various scalars across instability regimes (weakly to strongly convective).
2. Decomposing the vertical turbulent fluxes into main modes of (interpretable) variability associated with shear and convection.
3. Quantifying the diffusive part of each mode, its associated eddy diffusivity, and the non-local transport fraction.

The remainder of this work is structured as follows: In section 2, we thoroughly discuss the strategies and steps we take to develop our parameterization, providing justification for each step. Section 3 discusses our methodology, including data generation and preprocessing, as well as the neural network structure and training. In section 4, we present the results for flux prediction and their decomposition, followed by a discussion on projecting the flux onto a diffusing term in section 4.4. Finally, in section 5, we present our final discussion and conclusion.

2 Problem formulation and strategy

In this section, we provide a comprehensive outline of the steps and strategy we follow to parameterize and decompose the vertical turbulent fluxes.

First, as with most parameterizations of unresolved processes, our goal is to find a function that uses resolved quantities as input and predicts the unresolved physics. For the specific case of the dry convective boundary layer, we use the scalar and TKE pro-

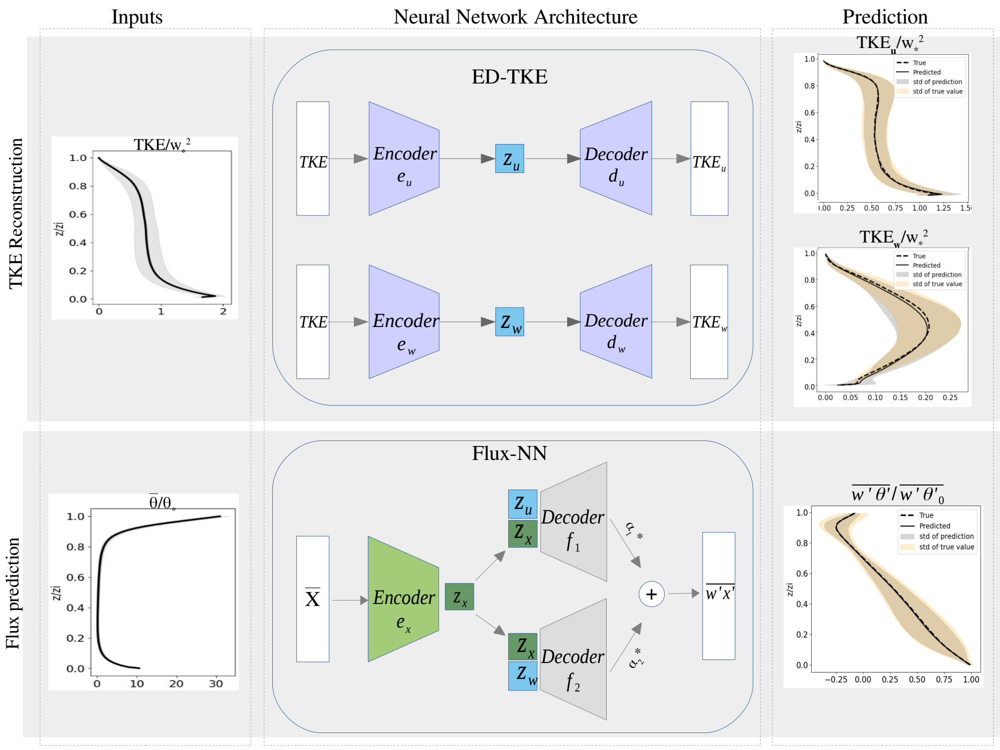


Figure 1: Neural network architecture.The model comprises two parts: ED-TKE and Flux-NN. In ED-TKE, two encoder-decoder units process turbulent kinetic energy (TKE) data, mapping it to lower-dimensional latent variables (z_u and z_w). These variables are then used by the decoders to predict the horizontal and vertical distribution of TKE. In Flux-NN, scalar profiles (e.g., heat, passive scalar) are mapped to a latent space (z_x), and the decoders combine the scalar's latent variables with those of TKE to predict the vertical turbulent flux of the corresponding scalar.

files as inputs to the neural network and aim to predict the vertical turbulent scalar flux as the target unresolved process. Mathematically, this can be expressed as follows:

$$\overline{w'x'} \approx \mathcal{F}(\overline{X}, TKE), \quad for\ any\ \overline{X} \tag{4}$$

$\mathcal{F}$ represents the mapping between a scalar and its vertical flux. Our goal is to learn a function capable of predicting the vertical turbulent flux for a diverse set of scalar profiles and across turbulent regimes. We rely on the neural network's capacity to approximate such a function, which allows us to diagnose turbulent fluxes, given the scalars and TKE profile, across various turbulent regimes and scalar profiles. The neural network's strength in capturing non-linear relationships between input and target variables makes this task achievable.

The approach of using the same function to parameterize various scalar profiles has already been widely employed in traditional parameterizations; for instance, EDMF and EDCG model heat and moisture flux in a convective boundary layer in a similar manner (Stull, 1988). More specifically, EDMF assumes a same formulation and equal eddy diffusivity and mass flux for moisture and heat. Therefore, any variations in the heat and moisture flux are attributed to differences in the moisture and heat profiles. It is worth noting that while this approximation of diagnosing all fluxes using the same function simplifies the modeling process, it does come at the cost of some accuracy. For instance, this approximation may not strictly hold in regions with strong stratification, such as in the inversion layer of the convective boundary layer, where gravity waves can potentially impact heat transport but not moisture or any passive scalars (Stull, 1976, 1973). Moreover, whether a scalar is passive or active can also affect the way it is transported by the flow. Nevertheless, approximating the fluxes of all scalars using the same function $\mathcal{F}$ and treating them similarly naturally constrains the solution space and $\mathcal{F}$ to be of much lower dimension, enabling the capture of relevant structures for prediction. Additionally, given the complexity of turbulent flows and the lack of comprehensive understanding of all the factors that may influence vertical fluxes, this assumption is often used as a reasonable approximation. Furthermore, since the goal is to develop a model that can be used in a variety of contexts and applications, we prioritize generality over strict accuracy. Finally, using multiple scalars with different profiles and sources/sinks and only one functional form, will reduce potential equifinalities.

To develop a more interpretable parameterization of the vertical turbulent flux of a scalar, we formulate the flux as the sum of two terms, or what we refer to as modes hereafter. Empirically, we have found that two modes are sufficient. In fact, decomposing the turbulent flux into more than two modes does not improve the accuracy of the parameterization; rather, it unnecessarily complicates and makes it less interpretable. While there is no strict mathematical justification for utilizing only two modes, it can be enforced by incorporating physical constraints into the flux decomposition, as is commonly done in most traditional parameterizations. For instance, by assuming a separation between local and non-local fluxes, EDMF and EDCG (Siebesma et al., 2007; J. Deardorff, 1972) decompose the total flux into two main modes. The Transport Asymmetry Approach (Moeng & Wyngaard, 1984, 1989) employs a different criterion and decomposes the total flux into contributions from top-down and bottom-up fluxes.

However, we do not employ a decomposition based on local-non-local or top-down-bottom-up flux, but rather enforce a dynamics-based decomposition. Our flux parameterization method involves decomposing the flux into two modes, where one mode represents the mechanically generated turbulence from wind shear, and the other mode represents the thermally generated turbulence from convection. By separating the contributions of these two modes, our method provides a more accurate representation of the physical processes involved in the turbulent flux. To achieve this, we use a large set of LES simulations with various wind shear and surface heating, thus a large range of turbulent regimes and train our neural network on all these simulations simultaneously. More importantly, we apply dimensionality reduction technique to the scalar and TKE profiles which allows us to capture the important structures in these profiles and their differences across turbulent regimes. Specifically, we observe that the shape of the TKE profile is heavily affected by the importance of wind shear versus surface heating and a well-designed encoder-decoder, when trained on a wide range of turbulent regimes, can effectively infer how much each process contributes into the TKE and thus the turbulent flux.

In a shear-driven boundary layer, where turbulence arises primarily from the interaction of wind shear with the flow, the horizontal TKE dominates, while vertical TKE is negligible. As the surface heat flux increases, thermally driven turbulence becomes important, and vertical TKE increases. Our preliminary results (not shown) unveil that the encoder-decoder, when applied to the TKE profile, captures information about the vertical and horizontal TKE into the latent space, which we then use to develop the flux decomposition. We discuss in detail the formulation and how we impose the constraint in section 3.3.

Therefore, we utilize the TKE and scalar profiles to create our vertical flux decomposition, which is formulated as follows:

Table 1: List of model parameters and some statistics averaged over one hour of simulation.

Name	Ug (ms^{-1})	$\overline{w'\theta'_0}$ (Kms^{-1})	$-z_i/L$	$w_*(ms^{-1})$	$u_*(ms^{-1})$)
Ug16 - $\overline{w'\theta'_0}$0.03	16	0.03	3.2	0.98	0.49
Ug16 - $\overline{w'\theta'_0}$0.06	16	0.06	6.1	1.26	0.51
Ug8 - $\overline{w'\theta'_0}$0.03	8	0.03	15.0	0.98	0.292
Ug4 - $\overline{w'\theta'_0}$0.05	4	0.05	302.8	1.17	0.128
Ug4 - $\overline{w'\theta'_0}$0.1	4	0.1	596.3	1.5	0.131
Ug2 - $\overline{w'\theta'_0}$0.1	2	0.1	1301	1.5	0.101

$$\overline{w'x'} = \alpha_1 f_1(\overline{X}, TKE) + \alpha_2 f_2(\overline{X}, TKE) \tag{5}$$

This equation assumes that each mode, represented by f_1 and f_2, depends on the scalar and TKE, with f_1 modeling shear-driven turbulence and f_2 modeling convective-driven turbulence. The coefficients α_1 and α_2 depend solely on large-scale forcing terms such as the geostrophic wind and surface heat flux and are learned through a neural network. We approximate f_1, f_2, α_1, and α_2 using a neural network, as described in detail in section 3.3.

3 Methodology

3.1 Data

We conduct six simulations using a large eddy simulation (LES) code developed by Albertson (1996) and Albertson and Parlange (1999). Validation of this model has been performed by Bou-Zeid et al. (2005) and V. Kumar et al. (2006). A detailed description of the numerical setup is provided in V. Kumar et al. (2006).

For subgrid-scale modeling, the LES uses a scale dependent Lagrangian model (Bou-Zeid et al., 2005) with a constant subgrid-scale Prandtl number of 0.4 for all scalars (Shah & Bou-Zeid, 2014). The domain is cubic with 256 grids in all three directions, with horizontal grid spacing of 24 meters and vertical spacing of 6 meters. The domain is doubly periodic in the horizontal direction, and the Coriolis parameter is set to $10^{-4}s^{-1}$. To prevent the reflection of gravity waves, LES has a sponge layer in the upper 25% of the domain. We set the initial potential temperature to 300 K below an initial PBL height ($z_i^0 = 0.8z_l$) and it increases with a lapse rate of 5K/km above this height.

We force all simulations with a constant surface heat flux $\overline{w'\theta'_0}$ and a constant pressure gradient expressed in terms of a geostrophic wind Ug in the x direction. These simulations represent a dry convective boundary layer with stability conditions ranging from weakly to strongly unstable. The stability parameter is defined as z_i/L, where z_i is the boundary layer height and L is the Obukhov length (Monin & Obukhov, 1954), defined as $u_*^3/[\kappa(g/T_0)\overline{w'\theta'_0}]$; u_* (ms^{-1}) is the surface friction velocity, and κ is the von Kármán constant. We run all simulations for 6-8 eddy turnovers, after which we record the instantaneous profiles every minute. Table 1 summarizes the settings for these simulations.

All simulations include three passive tracers with different initial and boundary conditions, which are used to better diagnose and disentangle the transport of updrafts, downdrafts and boundary layer top entrainment:

i) Surface-forced tracer ($\overline{S_{sf}}$) has a constant surface flux of 0.002 with no other sink or source in the domain. $\overline{S_{sf}}$ is initialized to zero throughout the domain. Figure 2.d and 2.i show the $\overline{S_{sf}}$ profile and its vertical flux, $\overline{w's'_{sf}}$, respectively.

ii) Entrainment-forced tracer ($\overline{S_{ef}}$) is initialized to zero below $0.8z_{i0}$ and to one above this level. The source of $\overline{S_{ef}}$ in the boundary layer is then only the intrusion of free tropospheric air with a high concentration of $\overline{S_{ef}}$ into the boundary layer via entrainment fluxes. Figure 2.e and 2.j show the $\overline{S_{ef}}$ profile and its vertical flux, $\overline{w's'_{ef}}$.

iii) Height-dependent tracer ($\overline{S_h}$) is initialized to $s(z, t = 0) = z/z_{i0}$. $\overline{S_h}$ has a constant relaxation term in its advection-diffusion equation that maintains its horizontal mean profile close to its initial profile. This relaxation term is $-\frac{s-s(t=t_0)}{\tau}$, where $\tau = \frac{z_i}{6}w_*$, following Q. Li et al. (2018). Figure 2.c and 2.h show the $\overline{S_h}$ profile and its vertical flux, $\overline{w's'_h}$.

In this paper, each simulation is identified using a naming convention that combines its geostrophic wind and surface heating. Specifically, we use a format of UgX-$\overline{w'\theta'}_0$Y, where X and Y represent the values of the geostrophic wind and surface heating, respectively. For instance, Ug16-$\overline{w'\theta'}_0$0.03 refers to a simulation with a geostrophic wind of 16 (ms^{-1}) and surface heating of 0.03 (Kms^{-1}). This naming convention is consistently used throughout the paper to refer to different simulations.

3.2 Prepossessing

3.2.1 Coarse-graining

To prepare the data for the neural network training, we coarse-grain the scalar snapshots to compute the state variables ($\overline{\theta}$, $\overline{TKE}$, $\overline{S_h}$, $\overline{S_{sf}}$, and $\overline{S_{ef}}$) and corresponding turbulent fluxes ($\overline{w'\theta'}$, $\overline{w's'_h}$, $\overline{w'e'}$, $\overline{w's'_{sf}}$, and $\overline{w's'_{ef}}$). The coarse-graining is only applied horizontally by averaging the data into larger grids. The averaging is based on a top-hat filter:

$$\overline{A}(i,j,k) = \frac{1}{L^2} \sum_{l=L(i-1)+1}^{l=Ni} \sum_{m=L(j-1)+1}^{m=Nj} A(l,m,k) \tag{6}$$

Here, A is the high-resolution field, N is the averaging factor, and i and j are indices in the x and y directions.

The fluxes are computed as follows:

$$\overline{w'x'} = \overline{wx} - \bar{w}\bar{x} \tag{7}$$

We coarse-grain the results presented here using $N = 64$ grids, roughly equal to 1.5 km. Given that the original horizontal domain is 256x256, this coarse-graining reduces the number of horizontal grids to 4x4. Taking into account the total number of snapshots for each simulation, this coarse-graining results in 20k samples of each scalar per simulation.

We simultaneously train the neural network on all scalars and simulations, based on our first assumption that all scalars are transported by turbulent flow in a similar way. Since we have six simulations and each simulation contains five scalars, the total number of samples is 6x5x20k, which equals 600k. We split these samples into training, validation, and test sets using a 70-10-20 percent ratio.

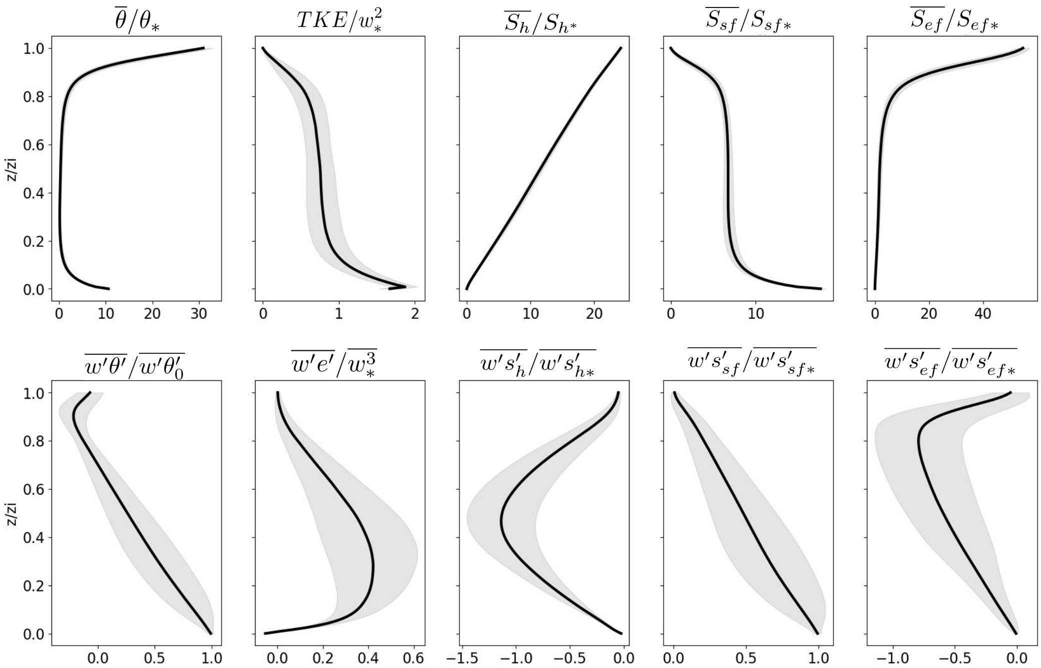


Figure 2: Inputs (shown in the first row) and outputs (shown in the second row) of the neural network.

3.2.2 Vertical interpolation

To train the NN, we use the entire column as input. However, we exclude the upper part of the simulation domain where the fluxes vanish, i.e., all layers above the top of the boundary layer (TOP). We define TOP as the height where the minimum of the second-order derivative of potential temperature occurs:

$$h_{top} \approx h(\min(\frac{d^2\overline{\theta}}{dz^2}))$$

Depending on the surface heat flux, TOP varies among simulations, which means that the number of layers between the surface and TOP is not the same for all simulations. This variation causes the dimension of the input to the NN to differ among simulations, which makes training with various input dimensions impractical. To address this challenge, we interpolate the same number of layers (128 layers) between the surface and the TOP for all simulations, thus standardizing the input dimension.

3.2.3 Non-dimensionalization

A proper scaling or non-dimensionalization of the inputs and outputs have been shown to improve the prediction and generalizability of a neural network (Beucler et al., 2021). To scale potential temperature, $\overline{\theta}$, and heat flux, $\overline{w'\theta'}$, we employ commonly used scaling parameters, θ_* and $\overline{w'\theta'_0}$, developed using the Buckingham–Pi theorem. For other variables we construct scaling parameters in a similar way done for θ_* and $\overline{w'\theta'_0}$. To scale a vertical turbulent flux (e.g., $\overline{w'x'}$), we divide it by a constant flux, which we show by $\overline{w'x'}^*$, as follows:

$$\overline{w'x'} \rightarrow \overline{w'x'}/\overline{w'x'}^*$$

The associated scalar of this flux is scaled by dividing the constant flux, $\overline{w'x'}_*$, by the Deardorff convective velocity scale, $w_* = (\frac{g}{T}\overline{w'\theta'_0}z_i)^{1/3}$ (J. W. Deardorff et al., 1970), the velocity scale for a convective boundary layer. We formulate this as:

$$\overline{X} \rightarrow \overline{X}/X_*, \quad where \;\; X_* = \overline{w'x'}_*/w_*$$

For the heat flux, $\overline{w'\theta'}$, we set $\overline{w'\theta'}_*$ to its surface value, $\overline{w'\theta'_0}$, which results in $X_* = \theta_*$. We scale the turbulent surface-forced tracer flux by its surface value $\overline{w's'_{sf_0}}$, while for other tracers, we choose a constant flux (e.g., the flux absolute maximum value) such that all turbulent scalar fluxes have comparable magnitudes.

3.3 Neural network

We use neural networks to model f_1, f_2, α_1, and α_2 to parameterize the vertical turbulent flux of scalars following Equation 5. However, rather than passing the high-dimensional profile of TKE and $\overline{X}$ directly to estimate f_1 and f_2 at each model level, we compress their profiles using non-linear dimensionality reduction techniques. This dramatically reduces the dimensionality of the f_1 and f_2 functions, and the number of degrees of freedom of the network. Using high resolution variables as input would result in an enormous degree of freedom, making it unlikely that a unique decomposition of fluxes can be achieved. Compressing the input allows us to capture the most important features of the data and model the fluxes with fewer parameters. This approach can also improve the model's efficiency and reduces the risk of overfitting, thereby improving the model's generalizability to new data. Further, non-linear dimensionality reduction techniques such as VAEs are particularly effective in capturing hidden structures in the data that are not immediately apparent in the high-dimensional input (Pu et al., 2016; Meng et al., 2017; Yang et al., 2019; Ma et al., 2020).

We perform flux prediction in two consecutive parts (Figure 1): in the first part, we train two separate encoder-decoders to predict horizontal and vertical TKE (hereafter TKE_u and TKE_w respectively) given TKE as input. Predicting TKE_u and TKE_w using encoder-decoders allows us to capture information related to these two variables directly from TKE in a latent space, which can be used for flux decomposition. Most climate models have a parameterization for TKE (i.e., first-order closure), but TKE_u and TKE_w are not separately available. We refer to this model as ED-TKE. In the second part of the flux retrieval, we employ an encoder-decoder network that receives the scalars profile alongside the low dimensional representation (latent space) of TKE_u and TKE_w from the first network, extracted from ED-TKE, and predict scalar flux (Figure 1, lower channel). We call this second sub-network NN-Flux. The two following subsections introduce the architecture of each neural network and discuss the underlying physical assumptions in detail.

3.3.1 Reconstructing TKE_u and TKE_w using double encoder-decoder

VAEs are deep learning models that consist of both an encoder and decoder. The encoder compresses high-dimensional input, such as the TKE profile in this case, into a low-dimensional latent space, and the decoder reverses this process by reconstructing the high-resolution input from its low-dimensional representation (Wang et al., 2014; Doersch, 2016). VAEs adopt a Bayesian perspective in the latent space and assume that the input to the second network, the encoder, is generated from a conditional probability distribution that describes an underlying generative model (Kingma & Welling, 2022). The multivariate, latent, representation of the input, typically denoted as $\mathbf{z}$, is assumed to follow a distribution $P(\mathbf{z})$. The model is then trained to maximize the probability of generating samples in the training dataset by optimizing both the reconstruction loss and the Kullback-Leibler divergence (KL divergence) of the approximate posterior, which is assumed to be Gaussian, as prior distribution. This Gaussian assumption is used so that the latent representation $\mathbf{z}$ can produce smooth and continuous reconstructions of the output, while trying to disentangle the different latent dimensions (as the Gaussian is assumed to be uncorrelated across dimensions and thus independent, as independence and uncorrelation are equivalent for Gaussian variables).

Most weather and climate atmospheric models have a prognostic equation for TKE but do not typically separate the horizontal and vertical TKE. Thus, we assume that TKE is available and can be used in the turbulent flux parameterization. As TKE consists of a horizontal and vertical part, it is desirable if its low dimension representation (z_{TKE}) can be first sub-partitioned to nodes representing horizontal TKE (hereafter z_u) and vertical TKE, hereafter z_w, separately. Based on (not shown) preliminary results, this partitioning is crucial for a proper and unambiguous flux decomposition in the second sub-network, where this latent representation (of TKE) is used to predict turbulent fluxes (see Figure 1). However, one challenge of using VAEs is that the disentanglement of latent variables is not guaranteed. Each latent variable may be a linear or nonlinear combination of the underlying latent representation, and this combination could vary among the profile. The entanglement of latent variables is a well-known issue in computer vision (Chen et al., 2018; Mathieu et al., 2019; Zietlow et al., 2021).

To address this disentanglement challenge, we use two encoder-decoder networks instead of the VAEs. The first network takes the TKE profile as input and predicts the horizontal component of TKE, TKE_u (upper branch), while the second network predicts the vertical component, TKE_w (lower branch). We refer to this combined model as ED-TKE for consistency with the previous naming convention. Unlike VAEs, these networks do not attempt to reconstruct the input from its low-dimensional representation; instead, they predict the horizontal and vertical components of TKE from the TKE profile itself. This is important because the aim of this network is not to learn a generative model but to decompose the TKE profile into its shear-driven (horizontal) and convective (vertical) components for use in the subsequent flux prediction step. To ensure that the low-dimensional representation of TKE is partitioned into separate nodes representing horizontal and vertical TKE (z_u and z_w, respectively), we use two separate encoder-decoder networks. The architecture of ED-TKE is shown in Figure 1. The ED-TKE function can be written mathematically as:

$$z_u = e_u(TKE) \tag{8a}$$
$$z_w = e_w(TKE) \tag{8b}$$
$$TKE_u = d_u(z_u) \tag{8c}$$
$$TKE_w = d_w(z_w) \tag{8d}$$

The encoder network e_u receives high-resolution (128 vertical levels) TKE profile and maps it to a low-dimensional representation, z_u. Similarly, e_w maps high-resolution TKE to z_w. The decoder networks d_u and d_w project z_u and z_w to high-resolution TKE_u and TKE_w, respectively. The objective (loss) function of ED-TKE is presented in Appendix A.

One important parameter in dimensionality reduction problems is the dimension of the latent space. Empirically, we find that when setting this dimension equal to two, the model demonstrates excellent performance in prediction. Increasing the dimension only leads to a more complex model that overfits and reproduces even small variabilities in the target outputs. Therefore, we set the dimension of both z_u and z_w to two. We use z_u and z_v as inputs to predict vertical turbulent fluxes.

We note that the horizontal and vertical TKE are interconnected and influenced by the flow, particularly at specific areas like the boundary of thermals where the rising and sinking air mixes and the conversion between two TKE terms are more prominent. However, since the proportion of these regions is relatively small and their effect on the corresponding TKE terms is minimal, we exclude these interactions in our flux decomposition. Additionally, our TKE-based decomposition is a first-order approximation, akin to PCA decomposition, where we assume that higher-order modes, which represent the interaction between the two forces, are negligible. Another option is to include higher-order modes that estimate the joint contribution of TKE_u and TKE_w to Equation 5 and construct a more complex approximation. However, this approach would require additional assumptions and constraints regarding the interaction between TKE_u and TKE_w, which are largely unknown and make the decomposition infeasible.

3.3.2 Predicting vertical turbulent flux

The second, bottom, module in Figure 1 depicts the architecture of the neural network that predicts the vertical turbulent fluxes. This model comprises an encoder, denoted by e_x, and two decoders, denoted by f_1 and f_2. The encoder, e_x, takes a high-dimensional scalar profile, $\overline{X}$, as input and encodes it to a low-dimensional latent space, hereafter referred to as z_x. The dimension of z_x is set to 2, as higher dimensions did not strongly improve the results yet became less interpretable.

$$z_x = e_x(\overline{X}) \tag{9}$$

where $\overline{X}$ represents the coarse-grained profile of any scalar, such as $\overline{\theta}$, $\overline{S_h}$, or $\overline{S_{sf}}$; thus:

$$\begin{aligned}
z_\theta &= e_x(\overline{\theta}/\theta_*) && \text{(10a)}\\
z_{s_h} &= e_x(\overline{S_h}/S_{h*}) && \text{(10b)}\\
z_{s_{sf}} &= e_x(\overline{S_{sf}}/S_{sf*}) && \text{(10c)}\\
z_{s_{ef}} &= e_x(\overline{S_{ef}}/S_{ef*}) && \text{(10d)}\\
z_e &= e_x(TKE/w_*^2) && \text{(10e)}
\end{aligned}$$

To predict fluxes, we utilize a neural network that incorporates Equation 5 (Figure 1. lower branch). We approximate f_1 and f_2 using two decoders and use the latent representation of scalar and TKE as the input to f_1 and f_2. This is in line with the discussion presented earlier.

For predicting the vertical turbulent flux of scalar X, we rewrite Equation 5 as:

$$\overline{w'x'} = \alpha_1 f_1(z_x, z_u) + \alpha_2 f_2(z_x, z_w) \tag{11}$$

By replacing $\overline{X}$ with various scalar profiles, we can represent their corresponding fluxes as follows:

$$\begin{aligned}
\overline{w'\theta'}/\overline{w'\theta'}_0 &= \alpha_1 f_1(z_\theta, z_u) + \alpha_2 f_2(z_\theta, z_w) && \text{(12a)}\\
\overline{w's'_h}/\overline{w's'}_{h*} &= \alpha_1 f_1(z_{s_h}, z_u) + \alpha_2 f_2(z_{s_h}, z_w) && \text{(12b)}\\
\overline{w's'_{sf}}/\overline{w's'}_{sf0} &= \alpha_1 f_1(z_{s_{sf}}, z_u) + \alpha_2 f_2(z_{s_{sf}}, z_w) && \text{(12c)}\\
\overline{w's'_{ef}}/\overline{w's'}_{ef*} &= \alpha_1 f_1(z_{s_{ef}}, z_u) + \alpha_2 f_2(z_{s_{ef}}, z_w) && \text{(12d)}\\
\overline{w'e'}/w_*^3 &= \alpha_1 f_1(z_e, z_u) + \alpha_2 f_2(z_e, z_w) && \text{(12e)}
\end{aligned}$$

The function e_x is used to map various scalar profiles to their corresponding latent representations (as described in Equation 10). These latent variables, along with z_u and z_w, are then passed to f_1 and f_2, which are shared across all scalar variables and used to predict the turbulent fluxes.

In order to complete our data-driven parameterization of the PBL fluxes, we must also model the two coefficients, α_1 and α_2, of the shear- and convective-dominated modes, in Equations 5 and 12. We further constrain these coefficients to be positive and to sum to unity, so they are a normalized weighting of each component:

$$\begin{aligned} \alpha_1 &> 0 \\ \alpha_2 &> 0 \\ \alpha_1 + \alpha_2 &= 1 \end{aligned}$$

These coefficients are predicted by a neural network with only large-scale conditions, $\overline{Ug}$ and $\overline{w'\theta'_0}$, serving as predictors. It is worth noting that it is only necessary to predict α_1. α_2 can then be computed as $\alpha_2 = 1 - \alpha_1$, following the third constraint listed above. The loss function of Flux-NN is discussed in Appendix A.

3.4 Training and validation

In this section, we describe our two-fold training process. First, we train the first module: the ED-TKE network to extract the latent variables of the TKE profile, z_u and z_w, which serve as inputs to the Flux-NN decoders. Subsequently, we train the second module: the Flux-NN model to predict the fluxes (Figure 1).

All encoders and decoders in both the ED-TKE and Flux-NN models consist of four hidden layers. The encoder layers have [128,64,32,16] neurons, while the decoder hidden layers have [16,32,64,128] neurons. Both networks take inputs in the form of mini-batches to train on an ensemble of small sampled profiles rather than individual samples. Each mini-batch consists of 128 samples drawn randomly from the various turbulent regimes and scalar profiles. Mini-batch training is a typical strategy for neural network optimization. The input shape to the encoders is $[n_{batch}, nz]$, where n_{batch} is the number of samples in each mini-batch, and n_z is the dimension of the coarse-grained profiles, which is 128, corresponding to the number of interpolated vertical levels. We train the model on mini-batches of 128 samples for 100 epochs, using early stopping with a patience of five epochs to prevent overfitting (Caruana et al., 2000). The networks are coded using TensorFlow (Abadi et al., 2016) and all hyperparameters (e.g., number of neurons in each layer, batch size) are tuned using the Sherpa library (Hertel et al., 2020).

At each iteration, the networks compute the loss averaged over the samples in one mini-batch, which contains samples from a diverse range of turbulent regimes, spanning strongly sheared to strongly convective flows. This loss value is then backpropagated through the network, and its derivative with respect to each NN parameter is computed. The NN parameters are then updated using the ADAM algorithm (Kingma & Ba, 2014). This process is repeated over all mini-batches, which correspond to one epoch. At the end of each epoch, the network's performance is validated using a validation dataset that the network has not seen during training. The training-validation process continues until either the total epochs are reached or an early stopping criteria are met. In this study, the early stopping criterion to minimize overfitting is based on the validation loss, and it has a patience of five epochs. This means that if the validation loss does not improve for five consecutive epochs, the network training stops. Early stopping is a powerful criterion for preventing network overfitting and achieving better generalization to unseen cases (Caruana et al., 2000).

To ensure the robustness of our results, we initialized the weights of each neural network randomly and ran ED-TKE with five different initializations. We also ran two randomly initialized Flux-NN for each ED-TKE run, resulting in a total of ten runs. The results are robust to random initialization of the network. The reported statistics, including R^2, are averaged across all runs, and the plots are generated using the run with the median R^2.

4 Results

4.1 ED-TKE

The ED-TKE network consists of two branches, each taking the TKE profile as input to its encoder. The top branch encodes the relevant information for predicting TKE_u into the two-dimensional latent variables z_{u_1} and z_{u_2}, while the bottom branch captures the information relevant for predicting TKE_w. The joint and marginal distributions of z_{u_1} and z_{u_2} are shown in Figure 3a, while Figure 3b shows the corresponding distributions for z_w. The marginal distribution of z_{w_1} is approximately Gaussian with similar mean and standard deviation across all simulations, which is enforced by the KL divergence term in the loss function (see Appendix A for more details). The latent variables z_u exhibit stronger non-Gaussian distribution and its distribution depends on the magnitude of geostrophic wind. Interestingly, some of the z_u variables have a bimodal marginal distribution, which deviates from the expected Gaussian distribution. This deviation can be attributed to the small weight assigned to the KL divergence term (KL_D) in the loss function (see Appendix A for details). The loss function of ED-TKE is a trade-off between achieving Gaussian-like marginal distributions and accurate predictions of TKE_u and TKE_w by the decoder. Increasing the weight of KL_D in the loss function may enforce Gaussianization of the marginal distributions, but it may also significantly decrease the accuracy of the predicted TKE_u and TKE_w. Since our model is focused on prediction rather than sample generation (with a stochastic latent space such as in variational auto-encoders), we decided to keep the weight of the KL divergence term small.

Figure 3c displays the predicted and true profiles of TKE_u, averaged over all samples from the same corresponding simulation across shear to convective regimes. The scaled TKE_u (divided by w_*^2) increases with the imposed wind and has the largest magnitude for the simulation Ug16-$\overline{w'\theta'}_0$0.03. The network's prediction of the TKE_u profile is highly similar to the true TKE_u for all simulations. This indicates that the TKE profile implicitly contains all the relevant information necessary for predicting TKE_u. By using an encoder, we can capture this information in a very low dimension, which can then be passed to a decoder to predict the horizontal TKE: TKE_u. In other words, having access to the total TKE profile in a model (such as a weather or climate model) is sufficient to implicitly uncover the split between horizontal TKE and vertical part of the total TKE, emphasizing that separate parameterizations for the horizontal and vertical TKEs might not be needed in the PBL.

The second branch of the ED-TKE network serves the same purpose as the first branch, but is specifically designed to predict the vertical TKE: TKE_w. Figure 3d demonstrates that TKE_w can also be accurately predicted from the TKE profile. In the convective boundary layer, TKE_w, normalized by w_*^2 and plotted as a function of z/zi, follows a universal parabolic shape that has been verified by laboratory experiments (Willis & Deardorff, 1974; R. Kumar & Adrian, 1986), measurements (Lenschow et al., 1980, 2012), and idealized simulations (J. W. Deardorff, 1974; Sullivan & Patton, 2011; Zhou et al., 2019). Our simulation results, as shown in Figure 3d, also confirm the existence of this universal profile. The predicted and true TKE_w profiles share the same overall

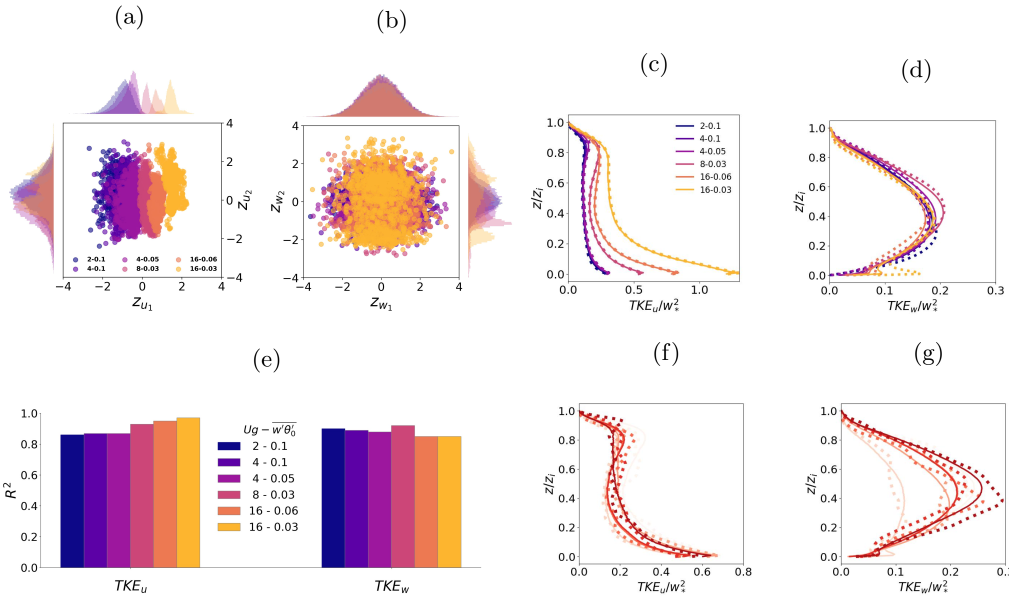


Figure 3: ED-TKE prediction: (a) displays the joint probability distribution of z_{u1} and z_{u2} extracted from the encoder trained on TKE profile. The marginal distributions are presented on the top (for z_{u1}) and the right side of the plot (for z_{u2}). (b) is similar to (a) but shows the joint probability distribution of z_w. Plot (c) displays the predicted (solid line) and true TKE_u (dashed line) averaged over each simulation, represented by colors. (d) is the same as (c) but for TKE_w. (e) shows the R^2 for TKE_u and TKE_w prediction. The colors represent different simulations, which are labeled in the legend as $Ug - \overline{w'\theta'_0}$. Finally, plots (f) and (g) respectively illustrate the networks' prediction (solid lines) and the true profiles (dashed lines) of TKE_u and TKE_w for randomly selected individual samples, distinguished by colors.

parabolic shape and primary peak. In simulations where the wind is strong (such as Ug16-$\overline{w'\theta'}_0$0.03 and Ug16-$\overline{w'\theta'}_0$0.06), a secondary peak in TKE_w near the surface is observed, which deviates slightly from the universal parabolic profile. However, our predicted TKE_w still exhibits this secondary peak, albeit with a smaller magnitude. The largest underestimation occurs for simulation Ug16-$\overline{w'\theta'}_0$0.03, where the predicted normalized secondary peak has a maximum of 0.1, while the true value is 0.18. We further emphasize that our networks are trained across regimes and are not targeting one specific regime, such as this mostly shear-driven mode.

To further investigate the ED-TKE skill in predicting TKE_u and TKE_w, we evaluate the predicted profiles for individual samples as shown in Figures 3f and 3g. These samples are randomly drawn from the test set. Although the mean profiles of TKE_u and TKE_w appear very smooth (Figures 3c and 3d), individual samples exhibit considerable variability (Figures 3f and 3g). The network captures the overall shape of each individual sample while smoothing out the small fluctuations observable in the true profiles. This behavior is consistent with existing literature (Takida et al., 2022) on the smoothness of encoder-decoder predictions and dimensionality reduction techniques. These methods only retain the information that is most relevant for the prediction, resulting in a smoother output. Also noted is that we did not include any information on horizontal neighboring cells in our network prediction, yet horizontal transport and variability in TKE, could lead to level-specific variations that cannot be captured by our strategy.

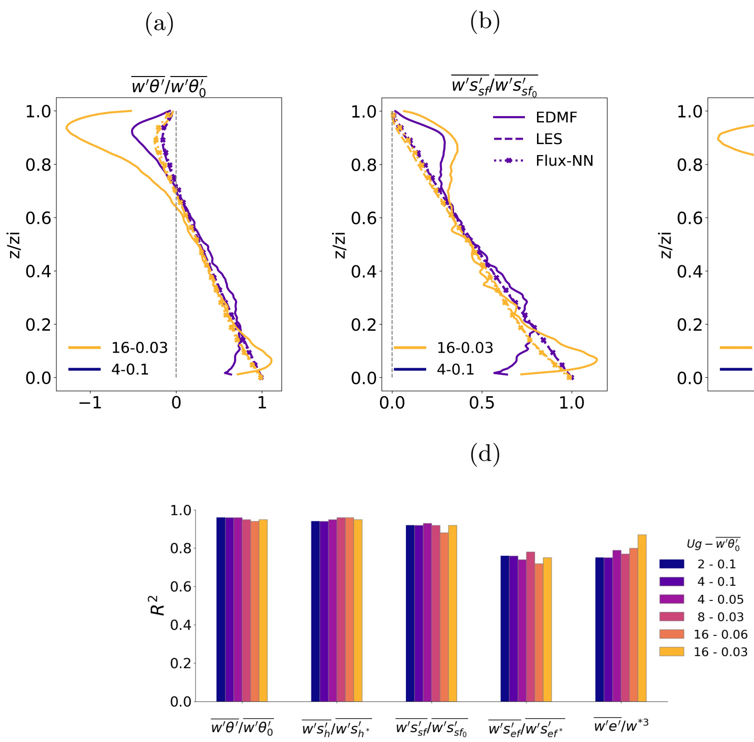


Figure 4: Plot shows the profiles of (a) vertical heat flux, (b) surface-forced tracer flux, and (c) entrainment-forced tracer flux, predicted by Flux-NN (dotted line), EDMF (solid lines), and computed from LES output (dashed line). Colors distinguish LES cases. Plot (d) shows R^2 computed for the neural network's prediction of turbulent fluxes for all simulations.

To quantify the skill of ED-TKE prediction, we compute R^2 for TKE_u and TKE_w and for each simulation separately (Figure 3e). R^2 is defined as one minus the ratio of the mean square error in prediction to the variance in the data. It ranges from zero to one, with one representing a perfect prediction with no error. For each simulation, we compute R^2 at each vertical level and then average layer-wise R^2 over all levels to obtain the final estimate (see Shamekh et al. (2022) for more detail). ED-TKE's prediction of TKE_u has a high R^2 (~ 0.9) across all simulations, while its prediction of TKE_w has a slightly lower R^2. Thus to summarize, our ED-TKE accurately captures relevant information for predicting TKE_u and TKE_w by only having access to TKE and shows a great performance across a large range of instability parameters present in the data set. We extract the latent variables from this network, z_u and z_v, to utilize as input for predicting vertical fluxes, as discussed in the next section.

4.2 Flux prediction

To predict the vertical turbulent fluxes of scalars and TKE, Flux-NN utilizes an encoder, e_x, to map the coarse-grained scalar or TKE profiles to a two-dimensional latent space (see Figure 1). These latent variables, along with z_u and z_w, are then processed by the decoders to predict the vertical turbulent flux profile of the corresponding scalar or TKE. In this section, we compare the Flux-NN predictions with fluxes directly computed from the coarse-grained LES output. Additionally, we compare our results with the reference ECMWF-implementation of EDMF scheme (Köhler et al., 2011), which has five tuning parameters. We re-tuned the EDMF parameters to obtain the best approximation of the heat flux for the Ug2-$\overline{w'\theta'}_0$0.1 run, which is most similar to the LES simulations utilized by Siebesma et al. (2007), in the originally developed parameterization. We subsequently use the re-tuned EDMF to predict the heat flux, surface-forced, and entrainment-forced tracer fluxes using their corresponding scalar profiles computed from our LES data (Figure 4).

The heat flux, normalized by its surface value, exhibits a universal profile as a function of normalized height z/z_i, decreasing linearly with height, reaching zero at the top of the mixed layer. In the inversion layer, the flux becomes negative and then approaches zero at the top of the boundary layer. Figure 4a illustrates the normalized turbulent heat fluxes predicted by Flux-NN (dotted lines), computed from LES outputs (dashed lines), and predicted by EDMF (solid lines) for two simulations one weakly and the other strongly unstable. The Flux-NN predictions closely match the coarse-grained fluxes computed from the LES for both illustrated cases (shear- or convectively-dominated) depicted in Figure 4 (and Figure S1). The EDMF scheme demonstrates reasonable heat flux prediction in the mixed layer, particularly for the strongly convective cases (as it was intended to). However, its prediction deviates from the LES output in the surface layer, exhibiting a considerable overestimation for the sheared cases (i.e., Ug16-$\overline{w'\theta'_0}$0.03). This overestimation decreases for cases with weak geostrophic wind, indicating the scheme's shortcomings in predicting fluxes for convective boundary layers with strong winds. Although we have discussed only two of the simulations for brevity, these findings are valid for our other simulations as well.

Remarkably, our Flux-NN accurately predicts the inversion layer heat flux across instability regimes (see Figure 4). The inversion layer flux presents a significant challenge for most traditional parameterizations, as it is strongly influenced by updrafts originating from the surface layer (Fedorovich et al., 2004), shear across the inversion (Pino et al., 2003, 2006; Pino & Vilà-Guerau De Arellano, 2008), and the entrainment of free tropospheric air into the boundary layer (Garcia & Mellado, 2014; Haghshenas & Mellado, 2019). Most traditional parameterizations do not explicitly incorporate the entrainment fluxes in their formulation and the entertainment is instead typically handled by the eddy-diffusion flux as in the EDMF, yet with important deviations. Indeed, as shown in Figure 4, the EDMF dramatically overestimates the magnitude of the heat flux in the inversion layer, particularly for the simulation with strong wind shear (e.i., Ug16-$\overline{w'\theta'}_0$0.03).

The Flux-NN is equally accurate in predicting the (normalized) surface-forced and entrainment-forced tracer fluxes, closely emulating the LES output (Figures 4b and 4c). This accuracy holds even in the inversion layer. However, EDMF significantly overestimates this part of the flux, particularly for entrainment-forced tracer, regardless of the geostrophic wind condition. This overestimation is related to the incorrect EDMF representation of the entrainment flux through the eddy diffusion. Given how important this entrainment is for key processes such as the diurnal growth of the PBL or shallow clouds formation and regimes, our new flux parameterization method might provide improvements to those key entrainment-related processes.

To further quantify the performance of Flux-NN, we computed the R^2 values separately for all simulations and fluxes (refer to Figure 4d). The R^2 values are very high (0.92-0.95) for $\overline{w'\theta'}$, $\overline{w's'_h}$, and $\overline{w's_{sf}}'$ across all simulations and turbulence regimes. However, for $\overline{w's_{ef}}'$ and $\overline{w'e'}$, the R^2 is smaller by about 0.1-0.15. Despite this, the flux prediction averaged over all samples of the same simulation is significantly close to the flux computed directly from the LES data for all scalars (Figure S1).

Additionally, to visualize the performance of Flux-NN at predicting individual samples, we randomly selected four samples for each scalar from the test data and plotted the predicted fluxes (solid lines) alongside the true fluxes (dashed lines) for these samples (Figure S1) with each sample distinguished by a different color. Despite the significant variability observed among samples of the same flux, particularly for $\overline{w's'_{ef}}$, $\overline{w's'_{h}}$, and $\overline{w'e'}$, Flux-NN accurately captures the overall shape of individual profiles while smoothing out fluctuations. This smoothing is similar to that observed in ED-TKE prediction and is related to the behavior of using reduced-order models, as discussed in section 4.1 and to the fact that we are not including the horizontal heterogeneity of the predictors in our vertical-only model. Thus, Flux-NN can predict vertical turbulent fluxes for various scalar profiles across a wide range of instability regimes, even in the inversion layer.

To summarize, Flux-NN accurately predicts turbulent fluxes of various scalars/TKE and provides a skillful approximation of all five fluxes across all six instability regimes (Figure 4d and Figure S1). Applying EDMF to the LES data reveals that this scheme does not generalize well to conditions with geostrophic winds or to tracers other than potential temperature. It overestimates the fluxes near the surface and in the inversion layer, particularly for entrainment-forced tracers, which rely heavily on the entrainment flux as the primary source of the scalar in the boundary layer. Additionally, the Flux-NN prediction of individual samples shows that the network can reproduce the overall shape of individual profiles while smoothing out fluctuations (Figure S1). This indicates that Flux-NN can predict the vertical turbulent fluxes of various scalars across a large range of instability regimes, even in the inversion layer. Therefore, it is a promising tool for modeling planetary boundary layers in climate and weather simulations.

4.3 Flux decomposition

The ED-TKE network discovers two separate latent variables that capture a hidden low-dimensional representation of horizontal and vertical TKE, which we refer to as z_u and z_w, respectively. The Flux-NN then utilizes these latent representations, along with z_x, to predict the contribution of each horizontal or vertical components to the total flux using Equation 12. We refer to each term in Equation 12 as a mode, with the first term ($\alpha_1 f_1(z_x, z_u)$) as the shear mode and the second term ($\alpha_2 f_2(z_x, z_w)$) as the convective mode. In this section, we discuss the shear and convective modes and their contributions to vertical turbulent fluxes, and investigate how this contribution changes across instability regimes. We primarily focus on turbulent heat, surface- and entrainment-forced tracer fluxes, while presenting results for TKE and height-dependent tracer fluxes in the supplementary material.

4.3.1 Vertical turbulent heat flux

Figure 5 illustrates the decomposition of the heat flux for all six simulations, with each mode normalized by its corresponding surface heat flux and plotted against the normalized height z/z_i. The shear mode (Figure 5b) is more prominent in simulations with a strong geostrophic wind, and its magnitude decreases as the instability parameter increases. In the most shear-driven simulation (e.g., Ug16-$\overline{w'\theta'}_0$0.03) the shear mode is responsible for approximately 80% of the total flux in the surface layer. Even in the mixed layer, the shear mode remains significant and explains about 70% of the flux. For the second most shear-driven simulation (e.g., Ug16-$\overline{w'\theta'}_0$0.06) and strongly convective cases, the contribution of the shear mode to the flux near the surface decreases from 75% and 50%, respectively. In these cases, the shear mode rapidly decreases with height, as expected, and becomes negligible in the mixed layer ($0.2 < z/z_i < 0.6$). In all simulations, the shear mode becomes negative in the upper part of the mixed layer ($z/z_i \sim 0.6 - 0.8$). In the inversion layer ($z/z_i \sim 0.8 - 1$), the shear mode increases (becomes more negative) with geostrophic wind, being more significant in highly sheared simulations.

Figure 5c depicts the convective modes of $\overline{w'\theta'}$ normalized by their respective surface heat flux and plotted as a function of z/z_i. The convective mode acts in the opposite direction to the shear mode and increases with instability, being larger for highly convective cases, as would be expected from basic understanding of the PBL. We note however that this behavior was not imposed but rather discovered by our networks when learning across simulation regimes. Despite differences in the instability parameters, the three most convective cases (Ug4-$\overline{w'\theta'}_0$0.05, Ug4-$\overline{w'\theta'}_0$0.1, and Ug2-$\overline{w'\theta'}_0$0.1) have very similar convective modes, which account for 50% of the flux near the surface and 100% in the mixed layer. Although one might expect the magnitude of the convective mode to increase with the PBL instability parameter, what we observe is that the convective mode is already quite large for Ug4-$\overline{w'\theta'}_0$0.05, which is in the free convective regime but has a smaller z_i/L compared to Ug4-$\overline{w'\theta'}_0$0.1 and Ug2-$\overline{w'\theta'}_0$0.1. Using quadrant analysis (Wyngaard & Moeng, 1992; D. Li & Bou-Zeid, 2011), Salesky et al. (2017) demonstrated that the heat transport efficiency also reaches a maximum past a given z_i/L threshold. Nonetheless, since their findings were based on quadrant analysis, we cannot make a direct comparison to our results.

In the inversion layer, the convective mode is strongest for simulations with larger instability parameters, thus Ug16-$\overline{w'\theta'}_0$0.03 and Ug16-$\overline{w'\theta'}_0$0.06 have the smallest contribution of convective mode into the flux in the inversion layer, and the three most unstable simulations have similar magnitudes.

The negative heat flux in the inversion layer has two sources: the overshoot of updrafts and the intrusion of free tropospheric air. The overshooting updrafts contain air with a negative θ anomaly and positive vertical velocity, thus creating a negative flux (Ghannam et al., 2017). On the other hand, the intrusion of free tropospheric air ventilates air with a positive θ anomaly and negative vertical velocity into the inversion layer, creating another negative heat flux. This intrusion is affected by the overshoot and wind shear in the inversion layer (Stull, 1976, 1973; Mcgrath-Spangler & Denning, 2010). Figure 5c suggests that the contribution of the convective mode to the inversion layer flux is larger for more convective cases, but it does not strongly scale with the surface heat flux or instability parameters. On the other hand, the intensity of the shear mode and its contribution to the inversion layer's flux depends on the strength of the wind shear. Thus, simulations Ug16-$\overline{w'\theta'}_0$0.03 and Ug16-$\overline{w'\theta'}_0$0.06 have the largest shear mode in the inversion layer. This finding is qualitatively consistent with that of Haghshenas and Mellado (2019); Garcia and Mellado (2014); Pino et al. (2003), showing the intensification of inversion layer flux with the wind shear.

4.3.2 Vertical turbulent surface-forced tracer flux

Figures 5e and 5f display the flux decomposition for the surface-forced tracer. The shear and convective modes of $\overline{w's_{sf}}$ highly resemble those of the turbulent heat flux, except in the inversion layer. The vertical flux of the surface-forced tracer is always positive, even in the inversion layer. This tracer has a source at the surface, and its concentration sharply decreases with height in the surface layer, then the tracer becomes nearly homogeneous vertically in the mixed layer (Figure 2). The surface-forced tracer concentration then rapidly decreases in the inversion layer, becoming zero in the free troposphere. The rising updrafts, which bring near-surface air with positive tracer anomaly into the inversion layer, create a positive flux. On the other hand, the entrainment flux injects free tropospheric air with a negative velocity and negative tracer anomaly (as they have a value of exactly zero above) into the inversion layer, generating a positive flux. Thus, the reduction of the surface-forced tracer concentration in the inversion layer results in its flux having the opposite sign of the heat flux one (Figure 5).

4.3.3 Vertical turbulent entrainment-forced tracer flux

Figure 2 shows the entrainment-forced tracer profile and its corresponding vertical turbulent flux computed from LES data, and Figure 5g shows the predicted flux for all simulations. Additionally, Figure 4c compares the predicted flux with the flux cal-

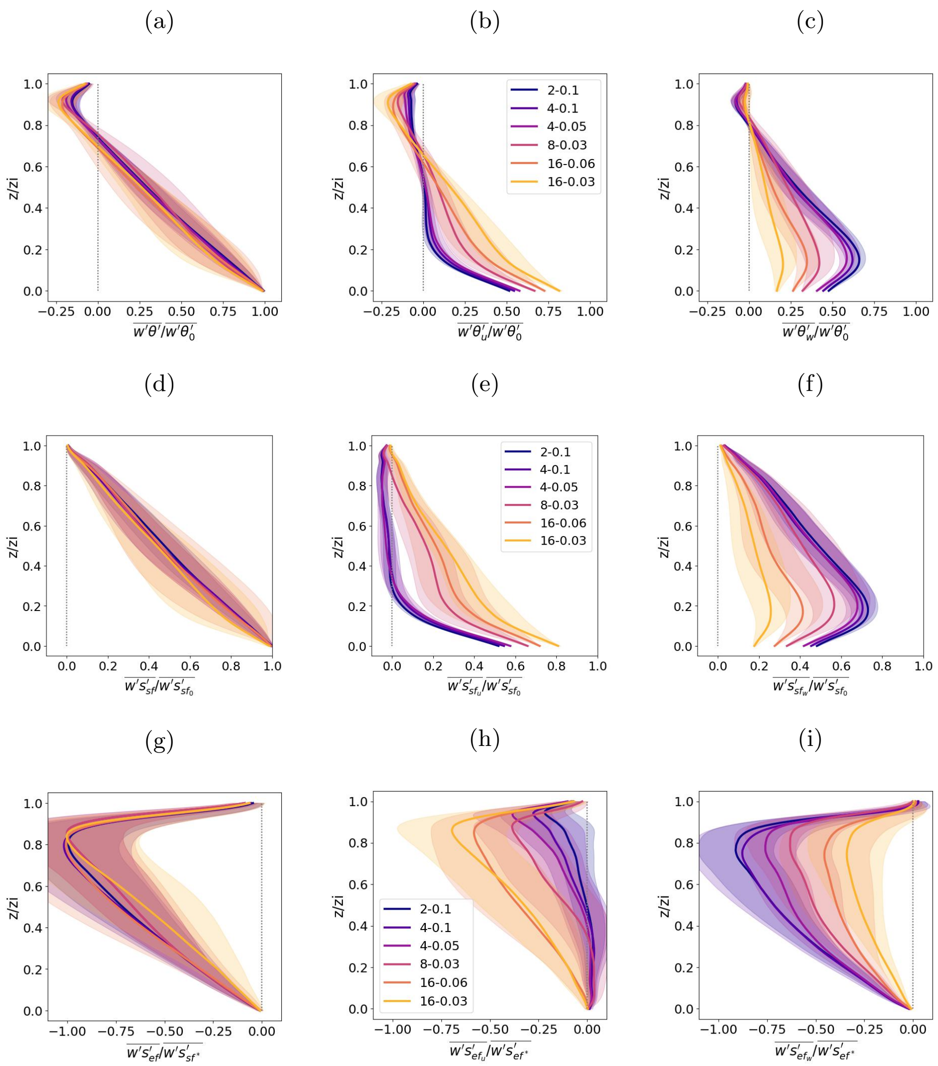


Figure 5: Plot shows (a) the vertical turbulent heat flux for various simulations, (b) shear mode represented as $\alpha_1 f_1$ in Equation 12.a, (b) convective mode represented as $\alpha_2 f_2$ in Equation 12.b, for heat flux decomposition. Plots d-f and g-i show the same as a-c but for surface-forced, and entrainment-forced tracer flux, respectively. The colors represent different simulations, which are labeled in the legend as $Ug - \overline{w'\theta'_0}$.

culated from LES data. This flux is negative across all six simulations. Figures 5h and 5i display shear and convective modes of the flux predicted by flux-NN. The shear mode of the strongly convective simulations is nearly zero from the surface to the middle of the mixed layer, at $z/z_i \sim 0.5$, indicating that the convective mode is mostly responsible for the flux at these layers. The significant contribution of the convective mode to the total flux highlights the importance of convective transport for the entrainment-forced tracer, despite the absence of a source near the surface or within the PBL. The only source of this tracer is the ventilation of free tropospheric air with a high tracer concentration into the boundary layer. Thus, the entrainment flux and downdraft play an essential role in this flux, bringing air with high tracer concentration downward, causing a negative flux. However, the updraft also contributes greatly to this flux by transporting near-surface air with a low tracer concentration upward, resulting in a negative flux. The role of the updraft in generating a vertical turbulent flux of entrainment-forced tracer, also known as top-down tracer, is often overlooked (Chor et al., 2020; Wyngaard & Brost, 1984). This is likely because the flux of this tracer can be fully explained by eddy-diffusivity models by assigning a large enough eddy diffusivity, as the flux is always down concentration gradient. Thus, since this tracer has no source near the surface, the role of updrafts in its flux is often disregarded (Chor et al., 2020). We show here that this is not the case. Our quadrant and subdomain-division analysis provide further confirmation of the significant contribution of updrafts and non-diffusive transport to the vertical turbulent flux of the entrainment-forced tracer (not shown).

In this section, we have discussed our approach of using a range of turbulent regimes, from shear-dominant to convective-dominant, to develop a constraint that enables us to decompose the total flux into two modes of variability. While there is no ground truth to accurately quantify our flux decomposition, we can qualitatively evaluate the two modes based on our physical understanding of turbulent flow and how the forcing can affect the flow. We also examined the flux decomposition for heat, surface- and entrainment-forced tracers and discussed the role of convective and shear modes in the vertical turbulent flux. Overall, the flux decomposition approach provides insight into the underlying mechanisms of turbulent flow and can be used to better understand and model the boundary layer dynamics.

4.4 Mode-specific estimation of diffusive flux using neural network

As mentioned in the introduction, most parameterizations of turbulent flux decompose the vertical turbulent flux into a diffusion and a non-diffusion term. Typically, the eddy diffusivity K needs to be parameterized, but there is no unique approach for doing so. Holtslag and Moeng (1991) define an eddy diffusivity using a simplified turbulent heat flux equation. This eddy diffusivity, which is related to the variance of vertical velocity, is adapted by Siebesma et al. (2007) for their EDMF scheme. Chor et al. (2020) estimate the diffusive and non-diffusive flux by maximizing for the diffusive part. Q. Li et al. (2021) employ a sub-domain decomposition approach and Taylor series expansion of the updraft and downdraft mass-flux transport to approximate down-gradient flux and then the eddy diffusivity. Lopez-Gomez et al. (2020) define an eddy mixing length based on constraints derived from the TKE balance.

While our TKE-based decomposition does not enforce a flux separation based on methods such as eddy length-scale or diffusivity, we are still interested in understanding the extent to which our extracted shear- and convective-modes exhibit diffusive behavior. To investigate this, we project each mode onto the vertical gradient of its corresponding scalar and determine the contribution of its diffusive part by maximizing the linear profile to the total flux. We use a regression neural network to predict an eddy diffusivity and compute the diffusive flux using Equation 1. As Figure B1 shows, for each vertical layer of the PBL, we calculate the vertical gradient of the scalars. Then, we input the TKE and the distance from the surface, z/z_i, of that layer into a neural network which outputs an eddy diffusivity value (K) for that specific layer. Next, we multiply K by the local gradient of the scalar (as per Equation 1) to estimate the total diffusive flux at that particular level. Although we do not have access to any ground truth value for the diffusive flux to use as a target value for supervised learning, we train the neural network to maximize the contribution of the diffusive flux to the total flux. In other words, we use our two modes f_1 and f_2 as the target value so that the network can predict an eddy diffusion flux that best matches these modes. Chor et al. (2020) used a similar approach to decompose the total flux into diffusive and non-diffusive components, but they predicted the entire vertical turbulent flux, whereas in our study, we project on each mode separately. This means that we determine the diffusive part of each mode, resulting in two eddy diffusivities, K_u and K_w, representing the eddy diffusivities of the shear and convective modes, respectively. We assume that these two K values are the

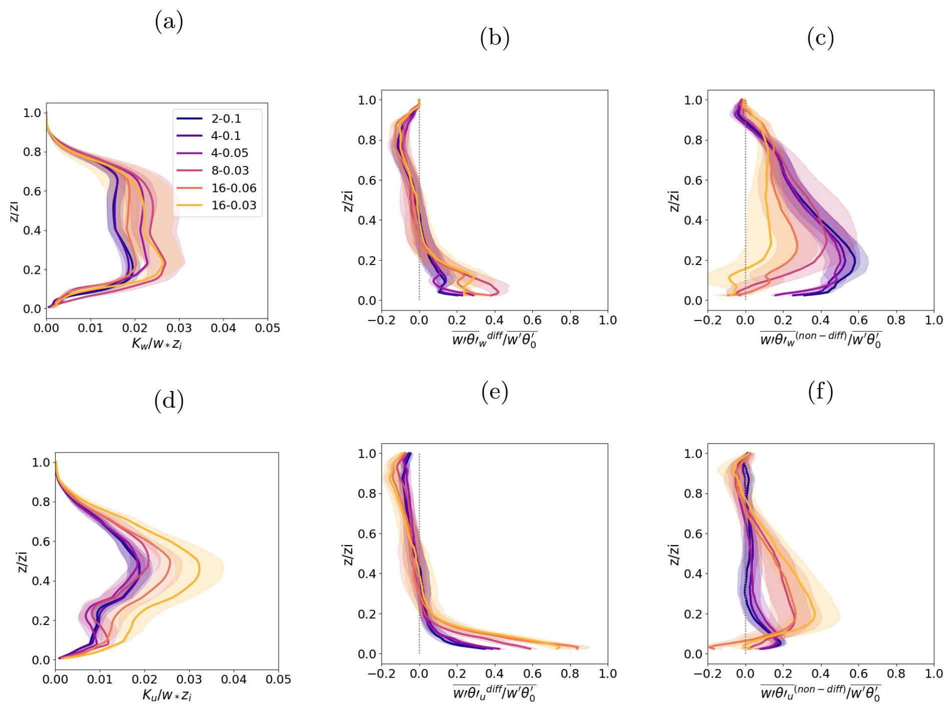


Figure 6: The plots depict the diffusive component of each mode of the vertical turbulent heat flux. In plot (a), the eddy diffusivity of the convective mode, denoted as K_w, is computed using a neural network. Plot (b) illustrates the diffusive portion of convective mode, while plot (c) shows the non-diffusive portion of the convective mode of the heat flux. Similarly, plots (d) to (f) present the corresponding information for the shear mode. The colors represent different simulations, which are labeled in the legend as $Ug - \overline{w'\theta'_0}$.

same for all scalars within the same simulation but vary across simulations. This assumption naturally constrains K_u and K_w, and we can express this projection as:

.

$$\overline{w'x'(z)}_w^{diff} = -NN_w(TKE_w(z), z/z_i).(\frac{\partial \overline{X}}{\partial z}(z)) \tag{14}$$

$$\overline{w'x'(z)}_u^{diff} = -NN_u(TKE_u(z), z/z_i).(\frac{\partial \overline{X}}{\partial z}(z)) \tag{15}$$

We use the neural network NN_w to predict the eddy diffusivity K_w and NN_u to predict K_u. After training the network and approximating the diffusive flux, we calculate the non-diffusive flux as a residual:

$$\overline{w'x'}_u^{Non-Diff} \sim \overline{w'x'}_u - (-K_u \frac{\partial \overline{X}}{\partial z})$$

for the shear mode and:

$$\overline{w'x'}_w^{Non-Diff} \sim \overline{w'x'}_w - (-K_w \frac{\partial \overline{X}}{\partial z})$$

for the convective mode. A detailed explanation of the neural network, its loss function, and the projection is provided in the Appendix B.

Figure 6a and 6d display the eddy diffusivity K_u and K_w normalized by $w_* z_i$, respectively, and plotted versus the normalized height z/z_i. To facilitate comparison with previously suggested eddy diffusivity, we plotted the eddy diffusivity computed based on Holtslag and Moeng (1991), hereafter K_H, shown in black lines in Figures S3, as a reference.

In Figure 6b and 6e, we present the diffusive parts of shear and convective mode, computed for the heat flux. The diffusive shear mode is significant in the surface layer but quickly diminishes to zero at approximately $z/z_i > 0.2$, and remains close to zero for $0.2 < z/z_i < 0.6$, where the vertical potential temperature gradient is insignificant. Therefore, a substantial portion of the shear mode, even for weakly convective cases, is non-diffusive (Figure 6f).

In the upper part of the mixed layer ($z/z_i > 0.6$), the diffusive shear flux becomes negative for both shear-driven and convective-driven cases. Interestingly, in the inversion layer, the shear mode is composed of both diffusive and non-diffusive components in shear-driven cases, but only the diffusive component is present in convective-driven cases. Similar to the shear mode, the convective mode (Figure 6b-c) is mostly non-diffusive except in the surface and inversion layers. In the inversion layer the diffusive convective mode is negative for all cases, and explains all convective mode flux.

Overall, we find that the two modes learned by the neural network are mostly non-diffusive, except in the surface and inversion layer. Additionally, the eddy diffusivity that we learn is about three times smaller than the eddy diffusivity suggested by Holtslag and Moeng (1991), as shown in Figure S3. The small magnitude of the diffusive flux implies that the Flux-NN model does not heavily rely on the diffusion term to predict the shear and convective modes. The model's latent variables can capture complex structures and learn both linear and non-linear relationships between scalars and fluxes, rather than just down-gradient ones.

Furthermore, when projecting the modes onto the scalar gradients, the neural network must simultaneously provide a down-gradient diffusive flux for all scalars, which places a stronger constraint on the magnitude of K. In other words, the diffusive flux must be down-gradient for all scalars, and learning an eddy diffusivity for only one scalar does not guarantee a down-gradient flux for a different scalar. Conventional parameterization often learns an eddy diffusivity term that compensates for neglected processes, such as down-draft or entrainment, resulting in an unrealistically large eddy diffusivity. This approach is commonly used in ocean mixed layer modeling.

5 Discussion and conclusion

To predict turbulent transport in the planetary boundary layer in numerical weather prediction and climate models, parameterizations have been widely adopted due to the models' limited spatial resolution. Historically, various approaches have been employed to parameterize turbulence, primarily based on scale separation, where separate schemes have been developed to represent small scale eddies and large scale coherent structures. In this work we focus on the dry convective boundary layer under different regimes from shear- to convective-dominated regimes and employ machine learning tools to develop a data-driven parameterization of vertical turbulent fluxes of various scalars and across a large range of instability regimes.

Although machine learning has become a popular tool for emulating physical processes, it faces two major issues: its high dimensionality that limits physical interpretability and therefore trust, and it typically lacks the integration of physical constraints into its emulators. In this work, we take a significant step towards solving these issues by introducing a lower-dimensional, latent representation of turbulent transport in the planetary boundary layer by introducing a physical constraint that enables us to decompose the flux into two main modes of variability. Our findings demonstrate that the latent representation of turbulent kinetic energy (TKE) can encode information related to the vertical and horizontal components of TKE, which reflect the relative contributions of thermal and mechanical turbulence to the vertical turbulent flux of a scalar. This is consistent with the fact that the turbulent flux in the boundary layer is primarily generated by the mechanical and buoyancy effects of wind shear and convection interacting with the flow, respectively. To ensure a separate representation of vertical and horizontal TKE in the latent space of TKE, we applied a physical constraint through the architecture of our neural network. Our approach involves using an encoder-decoder network that takes total TKE as input, which is readily available in most boundary layer parameterizations. By encapsulating the essential structural information needed for separately predicting horizontal and vertical TKE when given only total TKE as input, our network can effectively capture the relevant information for predicting these components. The TKE latent representation is then used to predict the vertical turbulent fluxes.

We showed that by reducing the dimension of TKE into two latent representations corresponding separately to horizontal and vertical TKE, we can accurately decompose the vertical flux of any scalar into two modes using a second set of neural networks. One of these modes is associated with horizontal TKE, which we refer to as a shear-driven mode, while the second mode is associated with vertical TKE and is called the convective mode. This flux decomposition is distinct from traditional schemes because it enables us to learn how each forcing contributes to the total flux and quantify their fractional contribution. By training the neural network on a wide range of scalars and simulations, we enable it to approximate a unique function for each mode that is independent of the scalar profile and turbulent regime. Additionally, these two modes and their variations with instability parameters are qualitatively consistent with our understanding of convection and shear contribution to the boundary layer vertical turbulent fluxes at various instability parameters.

Our analysis helps further refine our understanding of turbulent transport in the boundary layer and reveals that the neural network does not rely on the local gradient to generate the vertical turbulent fluxes. Specifically, by projecting each mode onto the gradient of its corresponding scalar, we observe that the fluxes are mostly non-diffusive, except in the surface and inversion layers. Even for entrainment-forced tracers, which exhibit fluxes down the gradient, the fluxes appear to be non-diffusive in our approach. In contrast, Chor et al. (2020) found that entrainment-forced tracer fluxes can be explained through diffusive fluxes even for the most convective case they studied. The contrasting results may stem from our neural network, which decomposes the flux without enforcing the gradient-following behavior, as opposed to their conventional diffusive approach. Our approach provides a unified framework to learn how each forcing contributes to the flux, offering insights into the underlying physical processes of turbulence in boundary layers.

We trained our neural network on a series of simulations, with instability parameters ranging from weakly unstable to strongly unstable. Our tests on the generalization of this network to unseen instability parameters indicate that the network exhibits skillful performance in interpolation. Specifically, when a simulation with an instability parameter between the minimum and maximum instability parameters present in the dataset is removed from the training set and used as a test set, the resulting R^2 value exceeds 0.8. Moreover, the network shows reasonable extrapolation capabilities when tested on cases with instability parameters larger than the range of instability parameters used in the training set. For example, when we remove the most convective simulation (Ug2-$\overline{w'\theta'}_0$0.1) from the training set and use it as a test set, the resulting R^2 value equals 0.75.

Hence, the model effectively extrapolates to unseen purely convective cases. This may be due to the fact that the non-dimensionalized profiles of TKE and scalars become similar at high instability parameters.

However, the network exhibits limitations in extrapolating to cases where the instability parameter is smaller than that of the training set. Removing the most shear driven simulation (Ug16-$\overline{w'\theta'}_0$0.03) from the training set and using it as a test set results in an R^2 value of 0.5. We attribute this shortcoming to the dynamics of the boundary layer turbulence, which become markedly different when the system approaches the neutral situation. Additionally, the non-dimensionalized fluxes and TKE profiles exhibit self-similarity for unstable simulations, leading to great extrapolation performance for both ED-TKE and flux-NN. However, for simulations with smaller instability parameters (i.e., near neutral turbulent regime), the non-dimensionalization does not result in a self-similar profile, making the extrapolation to simulations with instability parameters smaller than those in the training data much more challenging. In conventional parameterization of climate models, the three cases of stable, neutral, and convective conditions are often treated using three (or, in some cases, two) separate schemes, by switching from one scheme to another at a certain instability parameter which is, itself, set arbitrarily. This caveat is the subject of our future research to develop a parameterization that accurately models across a large range of instability parameters from strongly stable to strongly unstable situations.

One limitation of this study is the scale and grid dependency of our data-driven parameterization. Specifically, we coarse-grain the LES data to grids of 1.5x1.5 km^2, which lies within the "gray zone" of grid scales. Coarse-graining the data to a different grid size would alter the coarse profile of scalars and TKE, rendering the neural network trained on the original coarse data inaccurate for modeling other coarse data beyond the training set. In other words, our parameterization is not yet scale-adaptive. Furthermore, our network is trained on a specific vertical grid spacing and is, thus, sensitive to the grid spacing of the test data. Ideally, we aim to develop a model that is grid-agnostic such that it can be easily integrated into any weather or climate model, regardless of the horizontal grid size and vertical gird spacing used in the original data. We recognize this shortcoming and plan to address it in future research.

Appendix A Loss function

Variational Autoencoders (VAEs) take a Bayesian perspective and assume that the input to the encoder is generated from a conditional probability distribution that describes an underlying generative model. The multivariate latent representation of the input, denoted as z, is assumed to follow a prior distribution P(z). The model is then trained to maximize the probability of generating samples in the training dataset by optimizing both the reconstruction loss and the Kullback-Leibler divergence (KL divergence) of the approximate posterior, which is assumed to be Gaussian, from the prior distribution. Instead of predicting a single n-dimensional latent representation, the encoder predicts a mean and a standard deviation. The KL divergence term forces this distribution to be close to the prior distribution, which is typically assumed to be a normal distribution. This helps to enforce a disentanglement in the latent variables learned by the encoder, which is a property of interest in our work. Additionally, predicting a distribution instead of a single value results in a continuous latent space, which is valuable for using our neural network as a generator for parameterization. Therefore, we include the KL divergence in our loss.

We employ a variational encoder-decoder architecture, where we approximate the underlying generative model but instead of reconstructing the input TKE, we predict the horizontal and vertical TKE. Hence, our approach involves supervised training rather than unsupervised training. The loss consists of four terms: two are the mean squared errors of the predictions, and the other two are the KL divergences of the latent representations of the horizontal and vertical TKE.

The loss of predicting horizontal and vertical TKE is:

$$L_{MSE} = \frac{1}{N}\left(\sum_{i=1}^{N}\sum_{j=1}^{D}(TKE_u^t - TKE_u^p)_{ij}^2 + \sum_{i=1}^{N}\sum_{j=1}^{D}(TKE_w^t - TKE_w^p)_{ij}^2\right) \tag{A1}$$

where t represents the ground-truth coarse-grained profiles computed directly from LES, and p represents the coarse-grain profiles predicted by neural network. N represents the batch size and D is the dimension of the input which is 128.

The KL divergence loss, given the assumption of normal distribution for prior, is as follow

$$L_{KL_D} = \frac{1}{N} * \frac{1}{d}\left(\sum_{i=1}^{N}\sum_{k=1}^{d}(1 - ln\sigma_{u_{ik}}^2 + \mu_{u_{ik}}^2 + \sigma_{u_{ik}}^2) + \sum_{i=1}^{N}\sum_{k=1}^{d}(1 - ln\sigma_{w_{ik}}^2 + \mu_{w_{ik}}^2 + \sigma_{w_{ik}}^2)\right) \tag{A2}$$

where μ is the mean and σ is the standard deviation predicted by the encoder. d is the dimension of latent space, here equal to two and N is the batch size.

The total loss of ED-TKE is then the sum of the two terms:

$$loss_{ED} = L_{MSE} + \lambda L_{KL_D} \tag{A3}$$

λ is a hyperparameter that we empirically set to 10^{-1}. Assigning a lager value to λ increases the reconstruction error while assigning a smaller value reduces the Gaussianization of the distribution of the latent variables and their disentanglement. Gaussianization and disentanglement are desirable because many statistical models assume that the data is normally distributed, and by transforming the data to be closer to a Gaussian distribution, it can be easier to model and analyze the data. In the context of deep learning, Gaussianization can also help to regularize the learning process and prevent overfitting. Disentanglement refers to the property of the latent space where each dimension of the space represents a distinct and independent factor of variation in the data. This means that different aspects of the data are represented by different dimensions in the latent space, allowing for more precise manipulation and control of the data. Disentanglement can also help with interpretability and understanding of the model, as it provides a clear mapping between the latent space and the original data space. Therefore, by promoting Gaussianization and disentanglement in the latent space, we can improve the interpretability, flexibility, and generalization performance of the model.

The loss of Flux-NN is constructed the same way, by combining the KL divergence term with the MSE of flux prediction. This loss is then:

$$loss_{flux} = \frac{1}{N} * \frac{1}{D}\sum_{i=1}^{N}\sum_{j=1}^{D}(\overline{w'x'}^t_{ij} - \overline{w'x'}^p_{ij})^2 + \frac{1}{N} * \frac{1}{d}\sum_{i=1}^{N}\sum_{k=1}^{d}(1 - ln\sigma_{x_{ik}}^2 + \mu_{x_{ik}}^2 + \sigma_{x_{ik}}^2) \tag{A4}$$

Appendix B Predicting diffusive flux

Section 4.4 employs a neural network to predict the eddy diffusivity and, consequently, the diffusive component of each mode of variability of the turbulent fluxes. This appendix provides additional details on the network's architecture and its training process. Figure B1 displays the network's architecture and its associated loss function. The neural network takes layer-wise TKE and z/zi as inputs and generates a predicted value

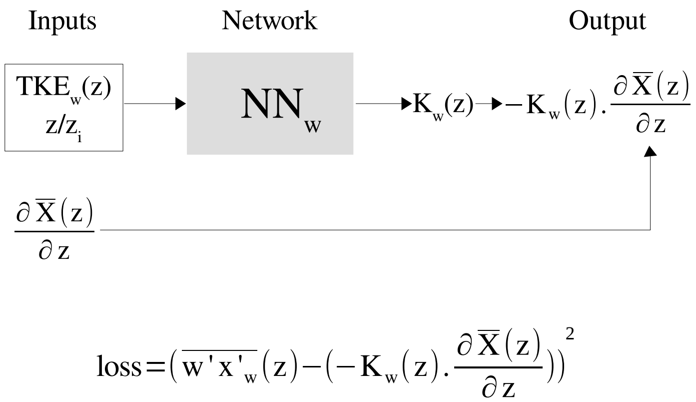


Figure B1: The neural network uses inputs such as $TKE_w(z)$ ($TKE_u(z)$) and z/z_i, representing the distance to the surface, to predict the eddy diffusivity $K_w(z)$ ($K_u(z)$). This eddy diffusivity is then multiplied by the scalar gradient to generate the output, which represents the diffusive flux. The network is trained with the target value of the convective (shear) mode, which compels the model to predict a diffusive flux as close as possible to the convective (shear) mode.

for eddy diffusivity. This predicted value is then multiplied by the gradient of the scalar, such as $\frac{\partial\bar{\theta}(z/zi)}{\partial z}$, resulting in the final prediction of the neural network. The network utilizes the convective (shear) mode as its target, meaning that it attempts to maximize the predicted diffusive component of each mode. This approach is similar to the one employed by (Chor et al., 2020), except that they did not use a neural network for their optimization.

The fully connected feed-forward neural network used in this study consists of four layers with 32, 64, 32, and 8 neurons in each layer, respectively. The final layer of the network, responsible for outputting the eddy diffusivity, employs a rectified linear unit (ReLU) activation function to ensure that the predicted eddy diffusivity remains positive. The network is trained using a batch size of 512 for 50 epochs, employing early stopping with a patience of five.

Open Research Section

The machine learning tools developed for this study as well as the scripts for pre- and post-processing data can be found here: `https://doi.org/10.5281/zenodo.8039033`

Acknowledgments

SS and PG acknowledge funding from European Research council grant USMILE, from Schmidt Future project M2LiNES and from the National Science Foundation Science and Technology Center (STC) Learning the Earth with Artificial intelligence and Physics (LEAP), Award # 2019625 - STC. SS is particularly grateful to Qi Li and Yuanfeng Cui for their generous assistance with large eddy simulations.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., . . . others (2016). Tensorflow: a system for large-scale machine learning. In *Osdi* (Vol. 16, pp.

265–283).
Albertson, J. D. (1996). *Large eddy simulation of land-atmosphere interaction.* University of California, Davis.
Albertson, J. D., & Parlange, M. B. (1999). Surface length scales and shear stress: Implications for land-atmosphere interaction over complex terrain. *Water Resources Research*, *35*(7), 2121-2132. doi: https://doi.org/10.1029/1999WR900094
Angevine, W. M., White, A. B., & Avery, S. K. (1994). Boundary-layer depth and entrainment zone characterization with a boundary-layer profiler. *Boundary-Layer Meteorology*, *68*(4), 375–385.
Bardina, J., Ferziger, J., & Reynolds, W. (1980). Improved subgrid-scale models for large-eddy simulation. In *13th fluid and plasmadynamics conference* (p. 1357).
Behrens, G., Beucler, T., Gentine, P., Iglesias-Suarez, F., Pritchard, M., & Eyring, V. (2022). Non-linear dimensionality reduction with a variational encoder decoder to understand convective processes in climate models. *Journal of Advances in Modeling Earth Systems*, *14*(8), e2022MS003130. (e2022MS003130 2022MS003130) doi: https://doi.org/10.1029/2022MS003130
Betts, A. (1973). Non-precipitating cumulus convection and its parameterization. *Quarterly Journal of the Royal Meteorological Society*, *99*(419), 178–196.
Beucler, T., Pritchard, M. S., Yuval, J., Gupta, A., Peng, L., Rasp, S., … Gentine, P. (2021). Climate-invariant machine learning. *CoRR*, *abs/2112.08440*.
Bolton, T., & Zanna, L. (2019). Applications of deep learning to ocean data inference and subgrid parameterization. *Journal of Advances in Modeling Earth Systems*, *11*(1), 376-399. doi: https://doi.org/10.1029/2018MS001472
Bou-Zeid, E., Meneveau, C., & Parlange, M. (2005). A scale-dependent lagrangian dynamic model for large eddy simulation of complex turbulent flows. *Physics of Fluids*, *17*(2), 025105. doi: 10.1063/1.1839152
Caruana, R., Lawrence, S., & Giles, C. (2000). Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. *Advances in neural information processing systems*, *13*.
Chen, R. T., Li, X., Grosse, R. B., & Duvenaud, D. K. (2018). Isolating sources of disentanglement in variational autoencoders. *Advances in neural information processing systems*, *31*.
Cheng, Y., Giometto, M., Kauffmann, P., Lin, L., Cao, C., Zupnick, C., … Gentine, P. (2019). Deep learning for subgrid-scale turbulence modeling in large-eddy simulations of the atmospheric boundary layer. *arXiv preprint arXiv:1910.12125*.
Chinita, M. J., Matheou, G., & Teixeira, J. (2018). A joint probability density–based decomposition of turbulence in the atmospheric boundary layer. *Monthly Weather Review*, *146*(2), 503 - 523. doi: 10.1175/MWR-D-17-0166.1
Chor, T., McWilliams, J. C., & Chamecki, M. (2020). Diffusive–nondiffusive flux decompositions in atmospheric boundary layers. *Journal of the Atmospheric Sciences*, *77*(10), 3479–3494.
Corrsin, S. (1975). Limitations of gradient transport models in random walks and in turbulence. In F. Frenkiel & R. Munn (Eds.), *Turbulent diffusion in environmental pollution* (Vol. 18, p. 25-60). Elsevier. doi: https://doi.org/10.1016/S0065-2687(08)60451-3
Deardorff, J. (1972). Theoretical expression for the countergradient vertical heat flux. *Journal of Geophysical Research*, *77*(30), 5900–5904.
Deardorff, J. W. (1974). Three-dimensional numerical study of turbulence in an entraining mixed layer. *Bound.-Layer Meteor.*, *7*, 199–226.
Deardorff, J. W., et al. (1970). Convective velocity and temperature scales for the unstable planetary boundary layer and for rayleigh convection. *J. atmos. Sci*, *27*(8), 1211–1213.

Doersch, C. (2016). Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*.

Ertel, H. (1942). Der vertikale turbulenz-wärmestrom in der atmosphäre. *Meteor. Z*, *59*, 250–253.

Fedorovich, E., Conzemius, R., & Mironov, D. (2004). Convective entrainment into a shear-free, linearly stratified atmosphere: Bulk models reevaluated through large eddy simulations. *Journal of the atmospheric sciences*, *61*(3), 281–295.

Garcia, J. R., & Mellado, J. P. (2014). The two-layer structure of the entrainment zone in the convective boundary layer. *Journal of the Atmospheric Sciences*, *71*(6), 1935–1955.

Gentine, P., Bellon, G., & van Heerwaarden, C. C. (2015). A closer look at boundary layer inversion in large-eddy simulations and bulk models: Buoyancy-driven case. *Journal of the atmospheric Sciences*, *72*(2), 728–749.

Gentine, P., Pritchard, M., Rasp, S., Reinaudi, G., & Yacalis, G. (2018). Could machine learning break the convection parameterization deadlock? *Geophysical Research Letters*, *45*(11), 5742–5751.

Ghannam, K., Duman, T., Salesky, S. T., Chamecki, M., & Katul, G. (2017). The non-local character of turbulence asymmetry in the convective atmospheric boundary layer. *Quarterly Journal of the Royal Meteorological Society*, *143*(702), 494-507. doi: https://doi.org/10.1002/qj.2937

Haghshenas, A., & Mellado, J. P. (2019). Characterization of wind-shear effects on entrainment in a convective boundary layer. *Journal of Fluid Mechanics*, *858*, 145–183. doi: 10.1017/jfm.2018.761

Han, J., Witek, M. L., Teixeira, J., Sun, R., Pan, H.-L., Fletcher, J. K., & Bretherton, C. S. (2016). Implementation in the ncep gfs of a hybrid eddy-diffusivity mass-flux (edmf) boundary layer parameterization with dissipative heating and modified stable boundary layer mixing. *Weather and Forecasting*, *31*(1), 341 - 352. doi: 10.1175/WAF-D-15-0053.1

Hertel, L., Collado, J., Sadowski, P., Ott, J., & Baldi, P. (2020). Sherpa: Robust hyperparameter optimization for machine learning. *SoftwareX*. (Also arXiv:2005.04048. Software available at: https://github.com/sherpa-ai/sherpa)

Holtslag, A., & Moeng, C.-H. (1991). Eddy diffusivity and countergradient transport in the convective atmospheric boundary layer. *Journal of the Atmospheric Sciences*, *48*(14), 1690–1698.

Kalina, E. A., Biswas, M. K., Zhang, J. A., & Newman, K. M. (2021). Sensitivity of an idealized tropical cyclone to the configuration of the global forecast system–eddy diffusivity mass flux planetary boundary layer scheme. *Atmosphere*, *12*(2). doi: 10.3390/atmos12020284

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P., & Welling, M. (2022). *Auto-encoding variational bayes.*

Köhler, M., Ahlgrimm, M., & Beljaars, A. (2011). Unified treatment of dry convective and stratocumulus-topped boundary layers in the ecmwf model. *Quarterly Journal of the Royal Meteorological Society*, *137*(654), 43–57.

Kumar, R., & Adrian, R. J. (1986). Higher order moments in the entrainment zone of turbulent penetrative thermal convection. *J. Heat Transfer*, *108*, 323–329.

Kumar, V., Kleissl, J., Meneveau, C., & Parlange, M. B. (2006). Large-eddy simulation of a diurnal cycle of the atmospheric boundary layer: Atmospheric stability and scaling issues. *Water Resources Research*, *42*(6). doi: https://doi.org/10.1029/2005WR004651

Lenschow, D. H., Lothon, M., Mayor, S. D., Sullivan, P. P., & Canut, G. (2012). A comparison of higher-order vertical velocity moments in the convective boundary layer from lidar with in situ measurements and large-eddy simulation. *Boundary-layer meteorology*, *143*, 107–123.

Lenschow, D. H., Wyngaard, J. C., & Pennell, W. T. (1980). Mean-field and second-moment budgets in a baroclinic, convective boundary layer. *Journal of Atmospheric Sciences*, *37*(6), 1313 - 1326. doi: https://doi.org/10.1175/1520-0469(1980)037⟨1313:MFASMB⟩2.0.CO;2

Li, D., & Bou-Zeid, E. (2011). Coherent structures and the dissimilarity of turbulent transport of momentum and scalars in the unstable atmospheric surface layer. *Boundary-Layer Meteorology*, *140*, 243–262.

Li, Q., Cheng, Y., & Gentine, P. (2021). Connection between mass flux transport and eddy diffusivity in convective atmospheric boundary layers. *Geophysical Research Letters*, *48*(8), e2020GL092073.

Li, Q., Gentine, P., Mellado, J. P., & McColl, K. A. (2018). Implications of nonlocal transport and conditionally averaged statistics on monin–obukhov similarity theory and townsend's attached eddy hypothesis. *Journal of the Atmospheric Sciences*, *75*(10), 3403–3431.

Lopez-Gomez, I., Cohen, Y., He, J., Jaruga, A., & Schneider, T. (2020). A generalized mixing length closure for eddy-diffusivity mass-flux schemes of turbulence and convection. *Journal of Advances in Modeling Earth Systems*, *12*(11), e2020MS002161. (e2020MS002161 10.1029/2020MS002161) doi: https://doi.org/10.1029/2020MS002161

Ma, X., Lin, Y., Nie, Z., & Ma, H. (2020). Structural damage identification based on unsupervised feature-extraction via variational auto-encoder. *Measurement*, *160*, 107811.

Mathieu, E., Rainforth, T., Siddharth, N., & Teh, Y. W. (2019). Disentangling disentanglement in variational autoencoders. In *International conference on machine learning* (pp. 4402–4412).

Mcgrath-Spangler, E., & Denning, S. (2010). Impact of entrainment from overshooting thermals on land–atmosphere interactions during summer 1999. *Tellus B: Chemical and Physical Meteorology*, *62*(5), 441–454.

Meng, Q., Catchpoole, D., Skillicom, D., & Kennedy, P. J. (2017). Relational autoencoder for feature extraction. In *2017 international joint conference on neural networks (ijcnn)* (pp. 364–371).

Moeng, C.-H., & Wyngaard, J. C. (1984). Statistics of conservative scalars in the convective boundary layer. *Journal of Atmospheric Sciences*, *41*(21), 3161–3169.

Moeng, C.-H., & Wyngaard, J. C. (1989). Evaluation of turbulent transport and dissipation closures in second-order modeling. *Journal of the Atmospheric Sciences*, *46*(14), 2311–2330.

Monin, A., & Obukhov, A. (1954). Basic laws of turbulent mixing in the atmosphere near the ground. vol. 24. *Tr Akad Nauk SSSR Geofiz Inst*, 163–187.

Mooers, G., Pritchard, M., Beucler, T., Ott, J., Yacalis, G., Baldi, P., & Gentine, P. (2021). Assessing the potential of deep learning for emulating cloud superparameterization in climate models with real-geography boundary conditions. *Journal of Advances in Modeling Earth Systems*, *13*(5), e2020MS002385. (e2020MS002385 2020MS002385) doi: https://doi.org/10.1029/2020MS002385

Mooers, G., Tuyls, J., Mandt, S., Pritchard, M., & Beucler, T. G. (2021). Generative modeling of atmospheric convection. In *Proceedings of the 10th international conference on climate informatics* (p. 98–105). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3429309.3429324

Park, S.-B., Gentine, P., Schneider, K., & Farge, M. (2016). Coherent structures in the boundary and cloud layers: Role of updrafts, subsiding shells, and environmental subsidence. *Journal of the Atmospheric Sciences*, *73*(4), 1789–1814.

Perezhogin, P., Zanna, L., & Fernandez-Granda, C. (2023). Generative data-driven approaches for stochastic subgrid parameterizations in an idealized ocean model. *arXiv preprint arXiv:2302.07984*.

Pino, D., de Arellano, J. V.-G., & Duynkerke, P. G. (2003). The contribution of shear to the evolution of a convective boundary layer. *Journal of the atmospheric sciences*, *60*(16), 1913–1926.

Pino, D., Jonker, H. J., Arellano, J. V.-G. d., & Dosio, A. (2006). Role of shear and the inversion strength during sunset turbulence over land: characteristic length scales. *Boundary-layer meteorology*, *121*, 537–556.

Pino, D., & Vilà-Guerau De Arellano, J. (2008). Effects of shear in the convective boundary layer: analysis of the turbulent kinetic energy budget. *Acta Geophysica*, *56*(1), 167.

Priestley, C. H. B., & Swinbank, W. (1947). Vertical transport of heat by turbulence in the atmosphere. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, *189*(1019), 543–561.

Pu, Y., Gan, Z., Henao, R., Yuan, X., Li, C., Stevens, A., & Carin, L. (2016). Variational autoencoder for deep learning of images, labels and captions. *Advances in neural information processing systems*, *29*.

Rasp, S., Pritchard, M. S., & Gentine, P. (2018). Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, *115*(39), 9684–9689.

Salesky, S. T., Chamecki, M., & Bou-Zeid, E. (2017). On the nature of the transition between roll and cellular organization in the convective boundary layer. *Boundary-layer meteorology*, *163*, 41–68.

Shah, S., & Bou-Zeid, E. (2014). Very-large-scale motions in the atmospheric boundary layer educed by snapshot proper orthogonal decomposition. *Boundary-Layer Meteorology*, *153*(3), 355–387.

Shamekh, S., Lamb, K. D., Huang, Y., & Gentine, P. (2022, oct). Implicit learning of convective organization explains precipitation stochasticity. *PNAS*. doi: 10.1002/essoar.10512517.1

Siebesma, A. P., & Cuijpers, J. (1995). Evaluation of parametric assumptions for shallow cumulus convection. *Journal of Atmospheric Sciences*, *52*(6), 650–666.

Siebesma, A. P., Soares, P. M. M., & Teixeira, J. (2007). A combined eddy-diffusivity mass-flux approach for the convective boundary layer. *Journal of the Atmospheric Sciences*, *64*(4), 1230 - 1248. doi: https://doi.org/10.1175/JAS3888.1

Siebesma, A. P., & Teixeira, J. (2000). An advection–diffusion scheme for the convective boundary layer: Description and 1d results. In *Preprints, 14th symp. on boundary layers and turbulence, aspen, co, amer. meteor. soc* (Vol. 133, p. 136).

Smagorinsky, J. (1963). General circulation experiments with the primitive equations: I. the basic experiment. *Monthly weather review*, *91*(3), 99–164.

Soares, P. M. M., Miranda, P. M. A., Siebesma, A. P., & Teixeira, J. (2004). An eddy-diffusivity/mass-flux parametrization for dry and shallow cumulus convection. *Quarterly Journal of the Royal Meteorological Society*, *130*(604), 3365-3383. doi: https://doi.org/10.1256/qj.03.223

Stensrud, D. J. (2009). *Parameterization schemes: keys to understanding numerical weather prediction models*. Cambridge University Press.

Stull, R. B. (1973). Inversion rise model based on penetrative convection. *Journal of Atmospheric Sciences*, *30*(6), 1092 - 1099. doi: https://doi.org/10.1175/1520-0469(1973)030⟨1092:IRMBOP⟩2.0.CO;2

Stull, R. B. (1976). Internal gravity waves generated by penetrative convection. *Journal of Atmospheric Sciences*, *33*(7), 1279–1286.

Stull, R. B. (1988). *An introduction to boundary layer meteorology* (Vol. 13). Springer Science & Business Media.

Sullivan, P. P., & Patton, E. G. (2011). The effect of mesh resolution on convective boundary layer statistics and structures generated by large-eddy simulation. *J. Atmos. Sci.*, *68*, 2395–2415.

Taira, K., Brunton, S. L., Dawson, S. T., Rowley, C. W., Colonius, T., McKeon,

B. J., ... Ukeiley, L. S. (2017). Modal analysis of fluid flows: An overview. *Aiaa Journal*, *55*(12), 4013–4041.

Takida, Y., Liao, W.-H., Lai, C.-H., Uesaka, T., Takahashi, S., & Mitsufuji, Y. (2022). Preventing oversmoothing in vae via generalized variance parameterization. *Neurocomputing*, *509*, 137–156.

Troen, I., & Mahrt, L. (1986). A simple model of the atmospheric boundary layer; sensitivity to surface evaporation. *Boundary-Layer Meteorology*, *37*(1), 129–148.

Wang, W., Huang, Y., Wang, Y., & Wang, L. (2014). Generalized autoencoder: A neural network framework for dimensionality reduction. In *Proceedings of the ieee conference on computer vision and pattern recognition workshops* (pp. 490–497).

Willis, G. E., & Deardorff, J. W. (1974). A laboratory model of the unstable planetary boundary layer. *J. Atmos. Sci.*, *31*, 1297–1307.

Wyngaard, J. C., & Brost, R. A. (1984). Top-down and bottom-up diffusion of a scalar in the convective boundary layer. *Journal of Atmospheric Sciences*, *41*(1), 102–112.

Wyngaard, J. C., & Moeng, C.-H. (1992). Parameterizing turbulent diffusion through the joint probability density. *Boundary-layer meteorology*, *60*(1), 1–13.

Wyngaard, J. C., & Weil, J. C. (1991). Transport asymmetry in skewed turbulence. *Physics of Fluids A: Fluid Dynamics*, *3*(1), 155–162.

Yang, Y., Zheng, K., Wu, C., & Yang, Y. (2019). Improving the classification effectiveness of intrusion detection by using improved conditional variational autoencoder and deep neural network. *Sensors*, *19*(11), 2528.

Yuval, J., & O'Gorman, P. A. (2020). Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions. *Nature communications*, *11*(1), 1–10.

Zhou, B., Sun, S., Sun, J., & Zhu, K. (2019). The universality of the normalized vertical velocity variance in contrast to the horizontal velocity variance in the convective boundary layer. *Journal of the Atmospheric Sciences*, *76*(5), 1437 - 1456. doi: https://doi.org/10.1175/JAS-D-18-0325.1

Zietlow, D., Rolinek, M., & Martius, G. (2021). Demystifying inductive biases for (beta-) vae based architectures. In *International conference on machine learning* (pp. 12945–12954).

Learning Atmospheric Boundary Layer Turbulence

S. Shamekh[1,2], P. Gentine[1,2]

[1]Department of Earth and Environmental Engineering, Columbia University, New York, NY, USA
[2]Center for Learning the Earth with Artificial Intelligence And Physics (LEAP), Columbia University, New York, NY, USA

Key Points:

- We propose a physics-informed machine learning technique to predict the vertical turbulent fluxes in the planetary boundary layer
- The vertical turbulent fluxes are decomposed into wind shear and convective modes and their contributions to flux generation are approximated
- The vertical turbulent fluxes exhibit a non-diffusive nature with the estimated eddy diffusivity significantly smaller than previous estimates

Corresponding author: Sara Shamekh, `ss6287@columbia.edu`

Abstract

Accurately representing vertical turbulent fluxes in the planetary boundary layer is vital for moisture and energy transport. Nonetheless, the parameterization of the boundary layer remains a major source of inaccuracy in climate models. Recently, machine learning techniques have gained popularity for representing oceanic and atmospheric processes, yet their high dimensionality often limits interpretability. This study introduces a new neural network architecture employing non-linear dimensionality reduction (encoder-decoder) to accurately predict vertical turbulent fluxes in a dry convective boundary layer. Our method utilizes the vertical profiles of turbulent kinetic energy and scalars as input to extract a physically constrained two-dimensional latent space, providing the necessary yet minimal information for accurate flux prediction. For this study, we obtained data by coarse-graining Large Eddy Simulations covering a broad spectrum of boundary layer conditions, ranging from weakly to strongly unstable. These regimes, driven by shear or buoyancy, are employed to constrain the latent space disentanglement, enhancing interpretability. By applying this constraint, we decompose the vertical turbulent flux of various scalars into two main modes of variability: one associated with wind shear and the other with convective transport. Our data-driven parameterization accurately predicts vertical turbulent fluxes (heat and passive scalars) across turbulent regimes, surpassing state-of-the-art schemes like the eddy-diffusivity mass flux scheme. By projecting each variability mode onto its associated scalar gradient, we estimate the diffusive flux and learn the eddy diffusivity. The diffusive flux is found to be significant only in the surface layer for both modes and becomes negligible in the mixed layer. The retrieved eddy diffusivity is considerably smaller than previous estimates used in conventional parameterizations, highlighting the predominant non-diffusive nature of transport.

Plain Language Summary

This study focuses on better understanding and predicting the movement of moisture and energy in the lower part of the Earth's atmosphere, called the planetary boundary layer. This is important as it directly impacts our ability to make accurate weather forecasts and model the climate. The study utilizes neural networks to analyze extensive data derived from computer simulations of the atmosphere. The objective is to extract meaningful insights from this complex data and facilitate accurate predictions. To achieve this, we employ an advanced form of neural networks, called encoder-decoder, that is a dimensionality reduction technique. This approach aims to distill the most crucial information from the data while maintaining simplicity and interpretability. Through this process, the neural network effectively reduces the data to two key factors influencing the movement of moisture and energy: wind shear (variations in wind speed and direction) and convective transport (movement resulting from heating and cooling). Overall, this study demonstrates that employing machine learning techniques can significantly advance our understanding and prediction of the intricate processes occurring in the atmosphere. This, in turn, leads to the development of more precise climate models and improved weather forecasts.

1 Introduction

In the planetary boundary layer (PBL), turbulence occurs over a wide range of scales, causing the mixing and transport of moisture, heat, momentum, and chemical scalars (Stull, 1988). An accurate representation of turbulent mixing is crucial for predicting many critical climate processes, such as low clouds, lower free tropospheric humidity and temperature, air-sea interaction, and more (Stensrud, 2009). Climate and weather models, which use a discretized spatiotemporal representation of the physical equations, cannot resolve scales smaller than their grid size. Therefore, these models rely on parameterization, an approximation of the impact of unresolved physical processes based on resolved quantities, such as turbulent mixing occurring at unresolved scales and transporting momentum, energy and scalars.

Traditionally, boundary layer turbulent mixing was first assumed to behave as a diffusion and therefore to be occurring down local gradient:

$$\overline{w'x'} = -K\frac{d\overline{X}}{dz} \tag{1}$$

Where $K(m^2s^{-1})$ is called the eddy diffusivity, w is the vertical velocity, and X represents a scalar variable that is being transported by the flow. Over-line indicates a horizontal averaging, and prime is the deviation from the spatial average: $x\prime = X - \overline{X}$.

Although simple and intuitive, this scheme fails to accurately predict the turbulent heat flux in the mixed layer of the convective boundary layer, where a zero or positive gradient of potential temperature coexists with finite and positive heat flux (Corrsin, 1975; Stull, 1988). This positive heat flux has been associated with the impact of large turbulent coherent structures, such as updrafts and downdrafts (Park et al., 2016), that are ubiquitous in the convective boundary layer and connect the surface layer to the top of the boundary layer by transporting heat and other variables upward, quickly within a model time step. Rising updrafts are accompanied by a descending counterpart in the convective boundary layer, and by a top-of-the-boundary layer entrainment flux occurring between the weakly turbulent stable stratification above the boundary layer and the convective layer (Fedorovich et al., 2004; Gentine et al., 2015). Large eddies traveling over large distances do not respect the eddy diffusion local gradient perspective, as these coherent structures bring non-locality to the turbulent fluxes.

Over the past few decades, several approaches have been proposed to correct the eddy-diffusion approach and include the effect of non-local eddies in turbulent flux parameterization, mainly considering the non-locality by adding a non-local term to the eddy diffusion (Ertel, 1942; Priestley & Swinbank, 1947). A few examples of such approaches are the eddy diffusivity – counter-gradient, hereafter EDCG, (J. Deardorff, 1972; Troen & Mahrt, 1986; Holtslag & Moeng, 1991), the transport asymmetry (Moeng & Wyngaard, 1984, 1989; Wyngaard & Brost, 1984; Wyngaard & Weil, 1991; Wyngaard & Moeng, 1992), or the eddy diffusivity – mass flux (Siebesma & Cuijpers, 1995; Siebesma & Teixeira, 2000; Siebesma et al., 2007), which is now widely used in weather and climate models. While a thorough review of the vertical turbulent parameterization is out of the scope of this work, we briefly discuss the eddy diffusivity – mass flux (EDMF, Siebesma et al. (2007)) approach since it is widely used and several EDMF versions have been developed and implemented in operational weather forecasts and climate models. Thus, we will use this as a benchmark to evaluate our parameterization for modeling vertical turbulent fluxes.

The EDMF model assumes that the total vertical flux of a scalar (e.g., heat, moisture) is due to the contribution of strongly convective updrafts, which cover a negligible horizontal fractional area, and a complementary slowly subsiding environment, with negligible vertical velocity. The total flux of scalar X can then be written as:

$$\overline{w'x'} = a_u\overline{w'x'}^u + (1-a_u)\overline{w'x'}^e + a_u(w_u - \overline{w})(X_u - X_e) \tag{2}$$

where u and e represent the updraft and environment, respectively. a_u is the updraft fractional area. w_u and $\overline{w}$ are the mean vertical velocity over the updraft and environment, and X_u and X_e are the corresponding mean scalar. Assuming a small fractional area coverage of the updrafts and a negligible vertical velocity in the environment, we can eliminate the first term on the RHS, approximate $\overline{w}$ to be zero, and replace X_e with $\overline{X}$. Thus Equation 2 reduces to:

$$\overline{w'x'} \approx \overline{w'x'}^e + a_uw_u(X_u - \overline{X}) \tag{3}$$

The first term on the RHS of Equation 3 is modeled using an eddy diffusivity (Equation 1) and the second term is the mass flux, non-local, contribution to total vertical turbulent flux, which was inspired by modeling of deep convection (Betts, 1973) .

Despite its successes in improving purely convective boundary layer parameterization compared to other approaches (e.g, pure ED or EDCG), EDMF still has important shortcomings. First, the EDMF decomposes the total flux into ED, modelling small scale eddies, and MF, modelling large scale updrafts. However, these two terms are not coupled in any systematic way, a theory for the relative partitioning between these two contributions does not exist, and a theory for an optimal scale at which the continuous spectrum of boundary layer eddies can be divided into small eddies and large thermals has not been established. Additionally, one of the main assumptions in deriving Equation 3 is that the updraft fractional area is negligible. However, recent studies (Q. Li et al., 2021; Chinita et al., 2018; Park et al., 2016) suggest a fractional area of 20-30 percent. Consequently, some of approximations made to derive the two-term Equation 3 does not hold accurately. For instance, the first term in the RHS of Equation 2 has been shown to be important and responsible for local fluxes in updrafts (Q. Li et al., 2021), or X_u may have a non-negligible impact on the domain mean value $\overline{X}$. Furthermore, the original EDMF schemes have been developed for a purely convective boundary layer (Siebesma et al., 2007; Soares et al., 2004), i.e., with small wind shear, thus EDMF poorly generalizes to situations driven by both wind and convection (Kalina et al., 2021). Some models, employ a hybrid scheme, such that, for weakly convective cases, they use EDCG and, at a certain instability threshold, they switch to EDMF (Han et al., 2016). However, this threshold is set arbitrarily and the switch between parameterizations appears quite ad hoc, and rather, a unified treatment of turbulence would be preferred.

In addition, one of the main pitfalls of the EDMF approach is its lack of explicit treatment of boundary layer top entrainment processes, which ventilate and mix air from the lower troposphere into the boundary layer. Entrainment significantly impacts the growth and structure of the PBL (Angevine et al., 1994), the evolution of mixed layer properties, surface fluxes, and the formation and maintenance of shallow clouds (Haghshenas & Mellado, 2019). However, EDMF does not explicitly take entrainment into account, which is potentially one reason for its shortcomings in accurately predicting turbulent fluxes at the top of the PBL and the exchange of PBL and lower troposphere. For instance, at the European Center for Medium Weather Forecast, entrainment is added (as a fraction of the surface buoyancy flux) as a diagnostic correction term to the EDMF model to obtain reasonable diurnal growth of the PBL. Additionally, wind shear strongly affects the entrainment flux and should be accounted for along with (dry) convection (Haghshenas & Mellado, 2019). Therefore, a more complete treatment of turbulence in the PBL is required, ideally one that can account for varying regimes from shear- to convectively-driven conditions and all forms of transport in the boundary layer, including eddies driven by shear or convection and entrainment at the top of the boundary layer.

Machine learning has proven to be a powerful tool for parameterizing subgrid-scale processes in the atmosphere and the ocean, particularly with the rise in popularity of neural networks (NNs) and deep learning as well as the explosion of high-resolution simulation data. In the field of atmosphere and ocean modeling, deep neural networks have shown significant potential in replacing traditional parameterizations of unresolved subgrid-scale processes (Gentine et al., 2018; Rasp et al., 2018; Mooers, Pritchard, et al., 2021; Yuval & O'Gorman, 2020; Bolton & Zanna, 2019; Shamekh et al., 2022; Perezhogin et al., 2023) due to their power in approximating a non-linear mapping between observed and unobserved quantities. Using ocean data, convolutional NNs have been shown to accurately predict subgrid-scale turbulent fluxes when trained on coarse-scale data (Bolton & Zanna, 2019), which could account for the spatial auto-correlation in the input data. In a similar vein, Cheng et al. (2019) used Direct Numerical Simulation (DNS) data of the planetary boundary layer to train a neural network that outperforms popular Large

Eddy Simulation (LES) schemes like the Smagorinsky (Smagorinsky, 1963) and Smagorinsky-Bardina (Bardina et al., 1980) turbulent flux models.

The work mentioned above showed promise in using neural networks in climate and weather models to replace traditional parameterization. One avenue that deserves more exploration is the use of interpretable machine learning models tailored to the problem of interest and including physical constraints, as they could unveil new understanding of the underlying physics. One such candidate could be a reduced order model (ROM) that relies on the fact that even high-dimensional complex flows often exhibit a few dominant modes of variability (Taira et al., 2017) that can provide coarse but key information about the flow. Encoder-decoder and variational auto-encoder (VAE) (Kingma & Welling, 2022) are powerful examples of ROM that map high-dimensional complex data to a low-dimensional latent representation. This latent representation captures the dominant modes of variability or structure in the data and because of its reduced dimension, can be much more interpretable. Mooers, Tuyls, et al. (2021) showed that VAEs could reconstruct velocity fields from a super-parameterized storm-resolving model. Additionally, they showed that the latent space could be categorized into different clusters, each representing a specific convection regime. Behrens et al. (2022) took this approach further and showed that VAE could reconstruct large-scale variables and map the latent variables to convection tendencies. They found that each latent variable represented a specific type or aspect of convection

In this work, we use encoder-decoder models and present a novel approach to data-driven parameterization of turbulence in the convective boundary layer, collapsing the complexity of turbulence into a few dimensions: the latent space. This latent space's dimensions are then disentangled using physical constraints based on the forcing of the boundary layer regimes: wind shear and surface heating. This constraint allows us to decompose the total flux of a scalar into two modes: one related to wind shear; the other related to convection. We use encoder-decoder models to approximate the latent representations of the scalars and Turbulent Kinetic Energy (TKE) profiles and then use these representations to predict the corresponding turbulent fluxes and modes of variability. Using this neural network, we aim to achieve the following objectives:

1. Predicting the vertical turbulent flux of various scalars across instability regimes (weakly to strongly convective).
2. Decomposing the vertical turbulent fluxes into main modes of (interpretable) variability associated with shear and convection.
3. Quantifying the diffusive part of each mode, its associated eddy diffusivity, and the non-local transport fraction.

The remainder of this work is structured as follows: In section 2, we thoroughly discuss the strategies and steps we take to develop our parameterization, providing justification for each step. Section 3 discusses our methodology, including data generation and preprocessing, as well as the neural network structure and training. In section 4, we present the results for flux prediction and their decomposition, followed by a discussion on projecting the flux onto a diffusing term in section 4.4. Finally, in section 5, we present our final discussion and conclusion.

2 Problem formulation and strategy

In this section, we provide a comprehensive outline of the steps and strategy we follow to parameterize and decompose the vertical turbulent fluxes.

First, as with most parameterizations of unresolved processes, our goal is to find a function that uses resolved quantities as input and predicts the unresolved physics. For the specific case of the dry convective boundary layer, we use the scalar and TKE pro-

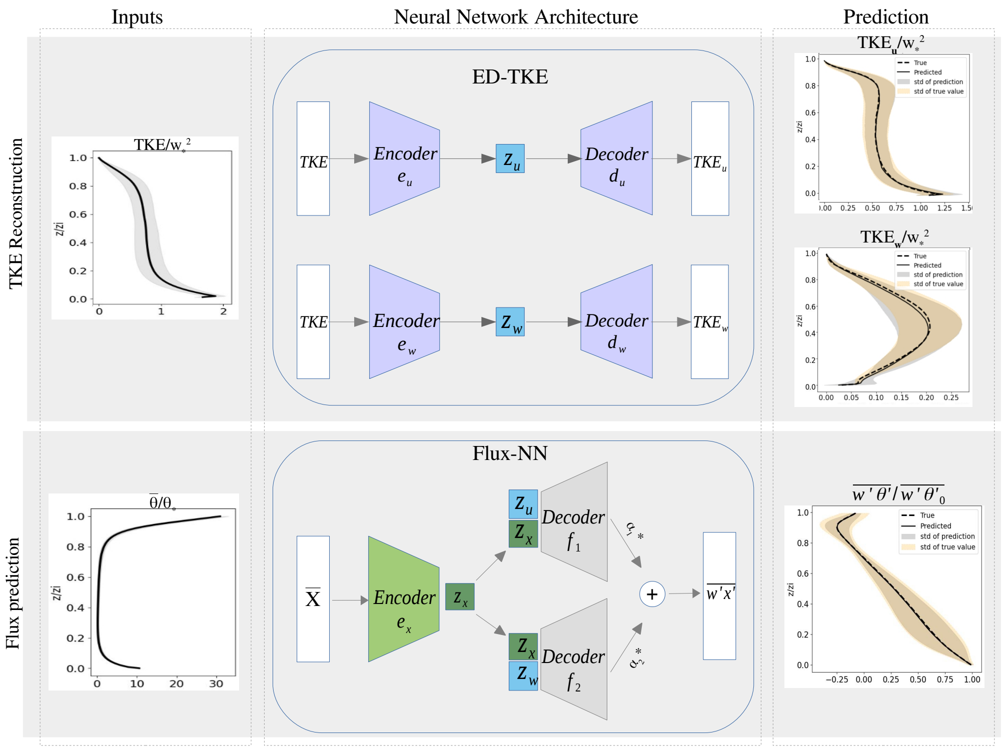


Figure 1: Neural network architecture.The model comprises two parts: ED-TKE and Flux-NN. In ED-TKE, two encoder-decoder units process turbulent kinetic energy (TKE) data, mapping it to lower-dimensional latent variables (z_u and z_w). These variables are then used by the decoders to predict the horizontal and vertical distribution of TKE. In Flux-NN, scalar profiles (e.g., heat, passive scalar) are mapped to a latent space (z_x), and the decoders combine the scalar's latent variables with those of TKE to predict the vertical turbulent flux of the corresponding scalar.

files as inputs to the neural network and aim to predict the vertical turbulent scalar flux as the target unresolved process. Mathematically, this can be expressed as follows:

$$\overline{w'x'} \approx \mathcal{F}(\overline{X}, TKE), \quad for\ any\ \overline{X} \tag{4}$$

$\mathcal{F}$ represents the mapping between a scalar and its vertical flux. Our goal is to learn a function capable of predicting the vertical turbulent flux for a diverse set of scalar profiles and across turbulent regimes. We rely on the neural network's capacity to approximate such a function, which allows us to diagnose turbulent fluxes, given the scalars and TKE profile, across various turbulent regimes and scalar profiles. The neural network's strength in capturing non-linear relationships between input and target variables makes this task achievable.

The approach of using the same function to parameterize various scalar profiles has already been widely employed in traditional parameterizations; for instance, EDMF and EDCG model heat and moisture flux in a convective boundary layer in a similar manner (Stull, 1988). More specifically, EDMF assumes a same formulation and equal eddy diffusivity and mass flux for moisture and heat. Therefore, any variations in the heat and moisture flux are attributed to differences in the moisture and heat profiles. It is worth noting that while this approximation of diagnosing all fluxes using the same function simplifies the modeling process, it does come at the cost of some accuracy. For instance, this approximation may not strictly hold in regions with strong stratification, such as in the inversion layer of the convective boundary layer, where gravity waves can potentially impact heat transport but not moisture or any passive scalars (Stull, 1976, 1973). Moreover, whether a scalar is passive or active can also affect the way it is transported by the flow. Nevertheless, approximating the fluxes of all scalars using the same function $\mathcal{F}$ and treating them similarly naturally constrains the solution space and $\mathcal{F}$ to be of much lower dimension, enabling the capture of relevant structures for prediction. Additionally, given the complexity of turbulent flows and the lack of comprehensive understanding of all the factors that may influence vertical fluxes, this assumption is often used as a reasonable approximation. Furthermore, since the goal is to develop a model that can be used in a variety of contexts and applications, we prioritize generality over strict accuracy. Finally, using multiple scalars with different profiles and sources/sinks and only one functional form, will reduce potential equifinalities.

To develop a more interpretable parameterization of the vertical turbulent flux of a scalar, we formulate the flux as the sum of two terms, or what we refer to as modes hereafter. Empirically, we have found that two modes are sufficient. In fact, decomposing the turbulent flux into more than two modes does not improve the accuracy of the parameterization; rather, it unnecessarily complicates and makes it less interpretable. While there is no strict mathematical justification for utilizing only two modes, it can be enforced by incorporating physical constraints into the flux decomposition, as is commonly done in most traditional parameterizations. For instance, by assuming a separation between local and non-local fluxes, EDMF and EDCG (Siebesma et al., 2007; J. Deardorff, 1972) decompose the total flux into two main modes. The Transport Asymmetry Approach (Moeng & Wyngaard, 1984, 1989) employs a different criterion and decomposes the total flux into contributions from top-down and bottom-up fluxes.

However, we do not employ a decomposition based on local-non-local or top-down-bottom-up flux, but rather enforce a dynamics-based decomposition. Our flux parameterization method involves decomposing the flux into two modes, where one mode represents the mechanically generated turbulence from wind shear, and the other mode represents the thermally generated turbulence from convection. By separating the contributions of these two modes, our method provides a more accurate representation of the physical processes involved in the turbulent flux. To achieve this, we use a large set of LES simulations with various wind shear and surface heating, thus a large range of turbulent regimes and train our neural network on all these simulations simultaneously. More importantly, we apply dimensionality reduction technique to the scalar and TKE profiles which allows us to capture the important structures in these profiles and their differences across turbulent regimes. Specifically, we observe that the shape of the TKE profile is heavily affected by the importance of wind shear versus surface heating and a well-designed encoder-decoder, when trained on a wide range of turbulent regimes, can effectively infer how much each process contributes into the TKE and thus the turbulent flux.

In a shear-driven boundary layer, where turbulence arises primarily from the interaction of wind shear with the flow, the horizontal TKE dominates, while vertical TKE is negligible. As the surface heat flux increases, thermally driven turbulence becomes important, and vertical TKE increases. Our preliminary results (not shown) unveil that the encoder-decoder, when applied to the TKE profile, captures information about the vertical and horizontal TKE into the latent space, which we then use to develop the flux decomposition. We discuss in detail the formulation and how we impose the constraint in section 3.3.

Therefore, we utilize the TKE and scalar profiles to create our vertical flux decomposition, which is formulated as follows:

Table 1: List of model parameters and some statistics averaged over one hour of simulation.

Name	Ug (ms^{-1})	$\overline{w'\theta'_0}$ (Kms^{-1})	$-z_i/L$	$w_*(ms^{-1})$	$u_*(ms^{-1}))$
Ug16 - $\overline{w'\theta'_0}$0.03	16	0.03	3.2	0.98	0.49
Ug16 - $\overline{w'\theta'_0}$0.06	16	0.06	6.1	1.26	0.51
Ug8 - $\overline{w'\theta'_0}$0.03	8	0.03	15.0	0.98	0.292
Ug4 - $\overline{w'\theta'_0}$0.05	4	0.05	302.8	1.17	0.128
Ug4 - $\overline{w'\theta'_0}$0.1	4	0.1	596.3	1.5	0.131
Ug2 - $\overline{w'\theta'_0}$0.1	2	0.1	1301	1.5	0.101

$$\overline{w'x'} = \alpha_1 f_1(\overline{X}, TKE) + \alpha_2 f_2(\overline{X}, TKE) \tag{5}$$

This equation assumes that each mode, represented by f_1 and f_2, depends on the scalar and TKE, with f_1 modeling shear-driven turbulence and f_2 modeling convective-driven turbulence. The coefficients α_1 and α_2 depend solely on large-scale forcing terms such as the geostrophic wind and surface heat flux and are learned through a neural network. We approximate f_1, f_2, α_1, and α_2 using a neural network, as described in detail in section 3.3.

3 Methodology

3.1 Data

We conduct six simulations using a large eddy simulation (LES) code developed by Albertson (1996) and Albertson and Parlange (1999). Validation of this model has been performed by Bou-Zeid et al. (2005) and V. Kumar et al. (2006). A detailed description of the numerical setup is provided in V. Kumar et al. (2006).

For subgrid-scale modeling, the LES uses a scale dependent Lagrangian model (Bou-Zeid et al., 2005) with a constant subgrid-scale Prandtl number of 0.4 for all scalars (Shah & Bou-Zeid, 2014). The domain is cubic with 256 grids in all three directions, with horizontal grid spacing of 24 meters and vertical spacing of 6 meters. The domain is doubly periodic in the horizontal direction, and the Coriolis parameter is set to $10^{-4}s^{-1}$. To prevent the reflection of gravity waves, LES has a sponge layer in the upper 25% of the domain. We set the initial potential temperature to 300 K below an initial PBL height ($z_i^0 = 0.8z_l$) and it increases with a lapse rate of 5K/km above this height.

We force all simulations with a constant surface heat flux $\overline{w'\theta'_0}$ and a constant pressure gradient expressed in terms of a geostrophic wind Ug in the x direction. These simulations represent a dry convective boundary layer with stability conditions ranging from weakly to strongly unstable. The stability parameter is defined as z_i/L, where z_i is the boundary layer height and L is the Obukhov length (Monin & Obukhov, 1954), defined as $u_*^3/[\kappa(g/T_0)\overline{w'\theta'_0}]$; u_* (ms^{-1}) is the surface friction velocity, and κ is the von Kármán constant. We run all simulations for 6-8 eddy turnovers, after which we record the instantaneous profiles every minute. Table 1 summarizes the settings for these simulations.

All simulations include three passive tracers with different initial and boundary conditions, which are used to better diagnose and disentangle the transport of updrafts, downdrafts and boundary layer top entrainment:

i) Surface-forced tracer ($\overline{S_{sf}}$) has a constant surface flux of 0.002 with no other sink or source in the domain. $\overline{S_{sf}}$ is initialized to zero throughout the domain. Figure 2.d and 2.i show the $\overline{S_{sf}}$ profile and its vertical flux, $\overline{w's'_{sf}}$, respectively.

ii) Entrainment-forced tracer ($\overline{S_{ef}}$) is initialized to zero below $0.8z_{i0}$ and to one above this level. The source of $\overline{S_{ef}}$ in the boundary layer is then only the intrusion of free tropospheric air with a high concentration of $\overline{S_{ef}}$ into the boundary layer via entrainment fluxes. Figure 2.e and 2.j show the $\overline{S_{ef}}$ profile and its vertical flux, $\overline{w's'_{ef}}$.

iii) Height-dependent tracer ($\overline{S_h}$) is initialized to $s(z,t=0) = z/z_{i0}$. $\overline{S_h}$ has a constant relaxation term in its advection-diffusion equation that maintains its horizontal mean profile close to its initial profile. This relaxation term is $-\frac{s-s(t=t_0)}{\tau}$, where $\tau = \frac{z_i}{6}w_*$, following Q. Li et al. (2018). Figure 2.c and 2.h show the $\overline{S_h}$ profile and its vertical flux, $\overline{w's'_h}$.

In this paper, each simulation is identified using a naming convention that combines its geostrophic wind and surface heating. Specifically, we use a format of UgX-$\overline{w'\theta'}_0$Y, where X and Y represent the values of the geostrophic wind and surface heating, respectively. For instance, Ug16-$\overline{w'\theta'}_0$0.03 refers to a simulation with a geostrophic wind of 16 (ms^{-1}) and surface heating of 0.03 (Kms^{-1}). This naming convention is consistently used throughout the paper to refer to different simulations.

3.2 Prepossessing

3.2.1 Coarse-graining

To prepare the data for the neural network training, we coarse-grain the scalar snapshots to compute the state variables ($\overline{\theta}$, $\overline{TKE}$, $\overline{S_h}$, $\overline{S_{sf}}$, and $\overline{S_{ef}}$) and corresponding turbulent fluxes ($\overline{w'\theta'}$, $\overline{w's'_h}$, $\overline{w'e'}$, $\overline{w's'_{sf}}$, and $\overline{w's'_{ef}}$). The coarse-graining is only applied horizontally by averaging the data into larger grids. The averaging is based on a top-hat filter:

$$\overline{A}(i,j,k) = \frac{1}{L^2} \sum_{l=L(i-1)+1}^{l=Ni} \sum_{m=L(j-1)+1}^{m=Nj} A(l,m,k) \tag{6}$$

Here, A is the high-resolution field, N is the averaging factor, and i and j are indices in the x and y directions.

The fluxes are computed as follows:

$$\overline{w'x'} = \overline{wx} - \bar{w}\bar{x} \tag{7}$$

We coarse-grain the results presented here using $N = 64$ grids, roughly equal to 1.5 km. Given that the original horizontal domain is 256x256, this coarse-graining reduces the number of horizontal grids to 4x4. Taking into account the total number of snapshots for each simulation, this coarse-graining results in 20k samples of each scalar per simulation.

We simultaneously train the neural network on all scalars and simulations, based on our first assumption that all scalars are transported by turbulent flow in a similar way. Since we have six simulations and each simulation contains five scalars, the total number of samples is 6x5x20k, which equals 600k. We split these samples into training, validation, and test sets using a 70-10-20 percent ratio.

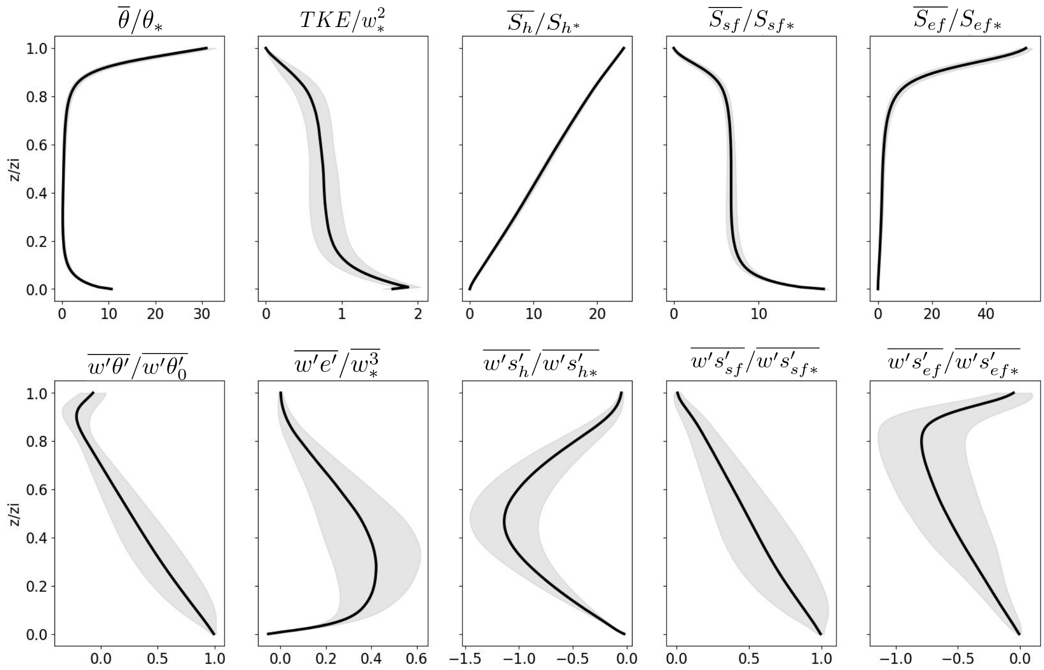


Figure 2: Inputs (shown in the first row) and outputs (shown in the second row) of the neural network.

3.2.2 Vertical interpolation

To train the NN, we use the entire column as input. However, we exclude the upper part of the simulation domain where the fluxes vanish, i.e., all layers above the top of the boundary layer (TOP). We define TOP as the height where the minimum of the second-order derivative of potential temperature occurs:

$$h_{top} \approx h(\min(\frac{d^2\overline{\theta}}{dz^2}))$$

Depending on the surface heat flux, TOP varies among simulations, which means that the number of layers between the surface and TOP is not the same for all simulations. This variation causes the dimension of the input to the NN to differ among simulations, which makes training with various input dimensions impractical. To address this challenge, we interpolate the same number of layers (128 layers) between the surface and the TOP for all simulations, thus standardizing the input dimension.

3.2.3 Non-dimensionalization

A proper scaling or non-dimensionalization of the inputs and outputs have been shown to improve the prediction and generalizability of a neural network (Beucler et al., 2021). To scale potential temperature, $\overline{\theta}$, and heat flux, $\overline{w'\theta'}$, we employ commonly used scaling parameters, θ_* and $\overline{w'\theta'_0}$, developed using the Buckingham–Pi theorem. For other variables we construct scaling parameters in a similar way done for θ_* and $\overline{w'\theta'_0}$. To scale a vertical turbulent flux (e.g., $\overline{w'x'}$), we divide it by a constant flux, which we show by $\overline{w'x'}^*$, as follows:

$$\overline{w'x'} \rightarrow \overline{w'x'}/\overline{w'x'}^*$$

The associated scalar of this flux is scaled by dividing the constant flux, $\overline{w'x'}_*$, by the Deardorff convective velocity scale, $w_* = (\frac{g}{T}\overline{w'\theta'_0}z_i)^{1/3}$ (J. W. Deardorff et al., 1970), the velocity scale for a convective boundary layer. We formulate this as:

$$\overline{X} \rightarrow \overline{X}/X_*, \quad where \ \ X_* = \overline{w'x'}_*/w_*$$

For the heat flux, $\overline{w'\theta'}$, we set $\overline{w'\theta'}_*$ to its surface value, $\overline{w'\theta'_0}$, which results in $X_* = \theta_*$. We scale the turbulent surface-forced tracer flux by its surface value $\overline{w's'_{sf_0}}$, while for other tracers, we choose a constant flux (e.g., the flux absolute maximum value) such that all turbulent scalar fluxes have comparable magnitudes.

3.3 Neural network

We use neural networks to model f_1, f_2, α_1, and α_2 to parameterize the vertical turbulent flux of scalars following Equation 5. However, rather than passing the high-dimensional profile of TKE and $\overline{X}$ directly to estimate f_1 and f_2 at each model level, we compress their profiles using non-linear dimensionality reduction techniques. This dramatically reduces the dimensionality of the f_1 and f_2 functions, and the number of degrees of freedom of the network. Using high resolution variables as input would result in an enormous degree of freedom, making it unlikely that a unique decomposition of fluxes can be achieved. Compressing the input allows us to capture the most important features of the data and model the fluxes with fewer parameters. This approach can also improve the model's efficiency and reduces the risk of overfitting, thereby improving the model's generalizability to new data. Further, non-linear dimensionality reduction techniques such as VAEs are particularly effective in capturing hidden structures in the data that are not immediately apparent in the high-dimensional input (Pu et al., 2016; Meng et al., 2017; Yang et al., 2019; Ma et al., 2020).

We perform flux prediction in two consecutive parts (Figure 1): in the first part, we train two separate encoder-decoders to predict horizontal and vertical TKE (hereafter TKE_u and TKE_w respectively) given TKE as input. Predicting TKE_u and TKE_w using encoder-decoders allows us to capture information related to these two variables directly from TKE in a latent space, which can be used for flux decomposition. Most climate models have a parameterization for TKE (i.e., first-order closure), but TKE_u and TKE_w are not separately available. We refer to this model as ED-TKE. In the second part of the flux retrieval, we employ an encoder-decoder network that receives the scalars profile alongside the low dimensional representation (latent space) of TKE_u and TKE_w from the first network, extracted from ED-TKE, and predict scalar flux (Figure 1, lower channel). We call this second sub-network NN-Flux. The two following subsections introduce the architecture of each neural network and discuss the underlying physical assumptions in detail.

3.3.1 Reconstructing TKE_u and TKE_w using double encoder-decoder

VAEs are deep learning models that consist of both an encoder and decoder. The encoder compresses high-dimensional input, such as the TKE profile in this case, into a low-dimensional latent space, and the decoder reverses this process by reconstructing the high-resolution input from its low-dimensional representation (Wang et al., 2014; Doersch, 2016). VAEs adopt a Bayesian perspective in the latent space and assume that the input to the second network, the encoder, is generated from a conditional probability distribution that describes an underlying generative model (Kingma & Welling, 2022). The multivariate, latent, representation of the input, typically denoted as $\mathbf{z}$, is assumed to follow a distribution $P(\mathbf{z})$. The model is then trained to maximize the probability of generating samples in the training dataset by optimizing both the reconstruction loss and the Kullback-Leibler divergence (KL divergence) of the approximate posterior, which is assumed to be Gaussian, as prior distribution. This Gaussian assumption is used so that the latent representation $\mathbf{z}$ can produce smooth and continuous reconstructions of the output, while trying to disentangle the different latent dimensions (as the Gaussian is assumed to be uncorrelated across dimensions and thus independent, as independence and uncorrelation are equivalent for Gaussian variables).

Most weather and climate atmospheric models have a prognostic equation for TKE but do not typically separate the horizontal and vertical TKE. Thus, we assume that TKE is available and can be used in the turbulent flux parameterization. As TKE consists of a horizontal and vertical part, it is desirable if its low dimension representation (z_{TKE}) can be first sub-partitioned to nodes representing horizontal TKE (hereafter z_u) and vertical TKE, hereafter z_w, separately. Based on (not shown) preliminary results, this partitioning is crucial for a proper and unambiguous flux decomposition in the second sub-network, where this latent representation (of TKE) is used to predict turbulent fluxes (see Figure 1). However, one challenge of using VAEs is that the disentanglement of latent variables is not guaranteed. Each latent variable may be a linear or nonlinear combination of the underlying latent representation, and this combination could vary among the profile. The entanglement of latent variables is a well-known issue in computer vision (Chen et al., 2018; Mathieu et al., 2019; Zietlow et al., 2021).

To address this disentanglement challenge, we use two encoder-decoder networks instead of the VAEs. The first network takes the TKE profile as input and predicts the horizontal component of TKE, TKE_u (upper branch), while the second network predicts the vertical component, TKE_w (lower branch). We refer to this combined model as ED-TKE for consistency with the previous naming convention. Unlike VAEs, these networks do not attempt to reconstruct the input from its low-dimensional representation; instead, they predict the horizontal and vertical components of TKE from the TKE profile itself. This is important because the aim of this network is not to learn a generative model but to decompose the TKE profile into its shear-driven (horizontal) and convective (vertical) components for use in the subsequent flux prediction step. To ensure that the low-dimensional representation of TKE is partitioned into separate nodes representing horizontal and vertical TKE (z_u and z_w, respectively), we use two separate encoder-decoder networks. The architecture of ED-TKE is shown in Figure 1. The ED-TKE function can be written mathematically as:

$$z_u = e_u(TKE) \tag{8a}$$
$$z_w = e_w(TKE) \tag{8b}$$
$$TKE_u = d_u(z_u) \tag{8c}$$
$$TKE_w = d_w(z_w) \tag{8d}$$

The encoder network e_u receives high-resolution (128 vertical levels) TKE profile and maps it to a low-dimensional representation, z_u. Similarly, e_w maps high-resolution TKE to z_w. The decoder networks d_u and d_w project z_u and z_w to high-resolution TKE_u and TKE_w, respectively. The objective (loss) function of ED-TKE is presented in Appendix A.

One important parameter in dimensionality reduction problems is the dimension of the latent space. Empirically, we find that when setting this dimension equal to two, the model demonstrates excellent performance in prediction. Increasing the dimension only leads to a more complex model that overfits and reproduces even small variabilities in the target outputs. Therefore, we set the dimension of both z_u and z_w to two. We use z_u and z_v as inputs to predict vertical turbulent fluxes.

We note that the horizontal and vertical TKE are interconnected and influenced by the flow, particularly at specific areas like the boundary of thermals where the rising and sinking air mixes and the conversion between two TKE terms are more prominent. However, since the proportion of these regions is relatively small and their effect on the corresponding TKE terms is minimal, we exclude these interactions in our flux decomposition. Additionally, our TKE-based decomposition is a first-order approximation, akin to PCA decomposition, where we assume that higher-order modes, which represent the interaction between the two forces, are negligible. Another option is to include higher-order modes that estimate the joint contribution of TKE_u and TKE_w to Equation 5 and construct a more complex approximation. However, this approach would require additional assumptions and constraints regarding the interaction between TKE_u and TKE_w, which are largely unknown and make the decomposition infeasible.

3.3.2 Predicting vertical turbulent flux

The second, bottom, module in Figure 1 depicts the architecture of the neural network that predicts the vertical turbulent fluxes. This model comprises an encoder, denoted by e_x, and two decoders, denoted by f_1 and f_2. The encoder, e_x, takes a high-dimensional scalar profile, $\overline{X}$, as input and encodes it to a low-dimensional latent space, hereafter referred to as z_x. The dimension of z_x is set to 2, as higher dimensions did not strongly improve the results yet became less interpretable.

$$z_x = e_x(\overline{X}) \tag{9}$$

where $\overline{X}$ represents the coarse-grained profile of any scalar, such as $\overline{\theta}$, $\overline{S_h}$, or $\overline{S_{sf}}$; thus:

$$\begin{aligned} z_\theta &= e_x(\overline{\theta}/\theta_*) && \text{(10a)}\\ z_{s_h} &= e_x(\overline{S_h}/S_{h*}) && \text{(10b)}\\ z_{s_{sf}} &= e_x(\overline{S_{sf}}/S_{sf*}) && \text{(10c)}\\ z_{s_{ef}} &= e_x(\overline{S_{ef}}/S_{ef*}) && \text{(10d)}\\ z_e &= e_x(TKE/w_*^2) && \text{(10e)} \end{aligned}$$

To predict fluxes, we utilize a neural network that incorporates Equation 5 (Figure 1. lower branch). We approximate f_1 and f_2 using two decoders and use the latent representation of scalar and TKE as the input to f_1 and f_2. This is in line with the discussion presented earlier.

For predicting the vertical turbulent flux of scalar X, we rewrite Equation 5 as:

$$\overline{w'x'} = \alpha_1 f_1(z_x, z_u) + \alpha_2 f_2(z_x, z_w) \tag{11}$$

By replacing $\overline{X}$ with various scalar profiles, we can represent their corresponding fluxes as follows:

$$\begin{aligned} \overline{w'\theta'}/\overline{w'\theta'}_0 &= \alpha_1 f_1(z_\theta, z_u) + \alpha_2 f_2(z_\theta, z_w) && \text{(12a)}\\ \overline{w's'_h}/\overline{w's'}_{h*} &= \alpha_1 f_1(z_{s_h}, z_u) + \alpha_2 f_2(z_{s_h}, z_w) && \text{(12b)}\\ \overline{w's'_{sf}}/\overline{w's'}_{sf0} &= \alpha_1 f_1(z_{s_{sf}}, z_u) + \alpha_2 f_2(z_{s_{sf}}, z_w) && \text{(12c)}\\ \overline{w's'_{ef}}/\overline{w's'}_{ef*} &= \alpha_1 f_1(z_{s_{ef}}, z_u) + \alpha_2 f_2(z_{s_{ef}}, z_w) && \text{(12d)}\\ \overline{w'e'}/w_*^3 &= \alpha_1 f_1(z_e, z_u) + \alpha_2 f_2(z_e, z_w) && \text{(12e)} \end{aligned}$$

The function e_x is used to map various scalar profiles to their corresponding latent representations (as described in Equation 10). These latent variables, along with z_u and z_w, are then passed to f_1 and f_2, which are shared across all scalar variables and used to predict the turbulent fluxes.

In order to complete our data-driven parameterization of the PBL fluxes, we must also model the two coefficients, α_1 and α_2, of the shear- and convective-dominated modes, in Equations 5 and 12. We further constrain these coefficients to be positive and to sum to unity, so they are a normalized weighting of each component:

$$\begin{aligned} \alpha_1 &> 0 \\ \alpha_2 &> 0 \\ \alpha_1 + \alpha_2 &= 1 \end{aligned}$$

These coefficients are predicted by a neural network with only large-scale conditions, $\overline{Ug}$ and $\overline{w'\theta'_0}$, serving as predictors. It is worth noting that it is only necessary to predict α_1. α_2 can then be computed as $\alpha_2 = 1 - \alpha_1$, following the third constraint listed above. The loss function of Flux-NN is discussed in Appendix A.

3.4 Training and validation

In this section, we describe our two-fold training process. First, we train the first module: the ED-TKE network to extract the latent variables of the TKE profile, z_u and z_w, which serve as inputs to the Flux-NN decoders. Subsequently, we train the second module: the Flux-NN model to predict the fluxes (Figure 1).

All encoders and decoders in both the ED-TKE and Flux-NN models consist of four hidden layers. The encoder layers have [128,64,32,16] neurons, while the decoder hidden layers have [16,32,64,128] neurons. Both networks take inputs in the form of mini-batches to train on an ensemble of small sampled profiles rather than individual samples. Each mini-batch consists of 128 samples drawn randomly from the various turbulent regimes and scalar profiles. Mini-batch training is a typical strategy for neural network optimization. The input shape to the encoders is $[n_{batch}, nz]$, where n_{batch} is the number of samples in each mini-batch, and n_z is the dimension of the coarse-grained profiles, which is 128, corresponding to the number of interpolated vertical levels. We train the model on mini-batches of 128 samples for 100 epochs, using early stopping with a patience of five epochs to prevent overfitting (Caruana et al., 2000). The networks are coded using TensorFlow (Abadi et al., 2016) and all hyperparameters (e.g., number of neurons in each layer, batch size) are tuned using the Sherpa library (Hertel et al., 2020).

At each iteration, the networks compute the loss averaged over the samples in one mini-batch, which contains samples from a diverse range of turbulent regimes, spanning strongly sheared to strongly convective flows. This loss value is then backpropagated through the network, and its derivative with respect to each NN parameter is computed. The NN parameters are then updated using the ADAM algorithm (Kingma & Ba, 2014). This process is repeated over all mini-batches, which correspond to one epoch. At the end of each epoch, the network's performance is validated using a validation dataset that the network has not seen during training. The training-validation process continues until either the total epochs are reached or an early stopping criteria are met. In this study, the early stopping criterion to minimize overfitting is based on the validation loss, and it has a patience of five epochs. This means that if the validation loss does not improve for five consecutive epochs, the network training stops. Early stopping is a powerful criterion for preventing network overfitting and achieving better generalization to unseen cases (Caruana et al., 2000).

To ensure the robustness of our results, we initialized the weights of each neural network randomly and ran ED-TKE with five different initializations. We also ran two randomly initialized Flux-NN for each ED-TKE run, resulting in a total of ten runs. The results are robust to random initialization of the network. The reported statistics, including R^2, are averaged across all runs, and the plots are generated using the run with the median R^2.

4 Results

4.1 ED-TKE

The ED-TKE network consists of two branches, each taking the TKE profile as input to its encoder. The top branch encodes the relevant information for predicting TKE_u into the two-dimensional latent variables z_{u_1} and z_{u_2}, while the bottom branch captures the information relevant for predicting TKE_w. The joint and marginal distributions of z_{u_1} and z_{u_2} are shown in Figure 3a, while Figure 3b shows the corresponding distributions for z_w. The marginal distribution of z_{w_1} is approximately Gaussian with similar mean and standard deviation across all simulations, which is enforced by the KL divergence term in the loss function (see Appendix A for more details). The latent variables z_u exhibit stronger non-Gaussian distribution and its distribution depends on the magnitude of geostrophic wind. Interestingly, some of the z_u variables have a bimodal marginal distribution, which deviates from the expected Gaussian distribution. This deviation can be attributed to the small weight assigned to the KL divergence term (KL_D) in the loss function (see Appendix A for details). The loss function of ED-TKE is a trade-off between achieving Gaussian-like marginal distributions and accurate predictions of TKE_u and TKE_w by the decoder. Increasing the weight of KL_D in the loss function may enforce Gaussianization of the marginal distributions, but it may also significantly decrease the accuracy of the predicted TKE_u and TKE_w. Since our model is focused on prediction rather than sample generation (with a stochastic latent space such as in variational auto-encoders), we decided to keep the weight of the KL divergence term small.

Figure 3c displays the predicted and true profiles of TKE_u, averaged over all samples from the same corresponding simulation across shear to convective regimes. The scaled TKE_u (divided by w_*^2) increases with the imposed wind and has the largest magnitude for the simulation Ug16-$\overline{w'\theta'}_0$0.03. The network's prediction of the TKE_u profile is highly similar to the true TKE_u for all simulations. This indicates that the TKE profile implicitly contains all the relevant information necessary for predicting TKE_u. By using an encoder, we can capture this information in a very low dimension, which can then be passed to a decoder to predict the horizontal TKE: TKE_u. In other words, having access to the total TKE profile in a model (such as a weather or climate model) is sufficient to implicitly uncover the split between horizontal TKE and vertical part of the total TKE, emphasizing that separate parameterizations for the horizontal and vertical TKEs might not be needed in the PBL.

The second branch of the ED-TKE network serves the same purpose as the first branch, but is specifically designed to predict the vertical TKE: TKE_w. Figure 3d demonstrates that TKE_w can also be accurately predicted from the TKE profile. In the convective boundary layer, TKE_w, normalized by w_*^2 and plotted as a function of z/zi, follows a universal parabolic shape that has been verified by laboratory experiments (Willis & Deardorff, 1974; R. Kumar & Adrian, 1986), measurements (Lenschow et al., 1980, 2012), and idealized simulations (J. W. Deardorff, 1974; Sullivan & Patton, 2011; Zhou et al., 2019). Our simulation results, as shown in Figure 3d, also confirm the existence of this universal profile. The predicted and true TKE_w profiles share the same overall

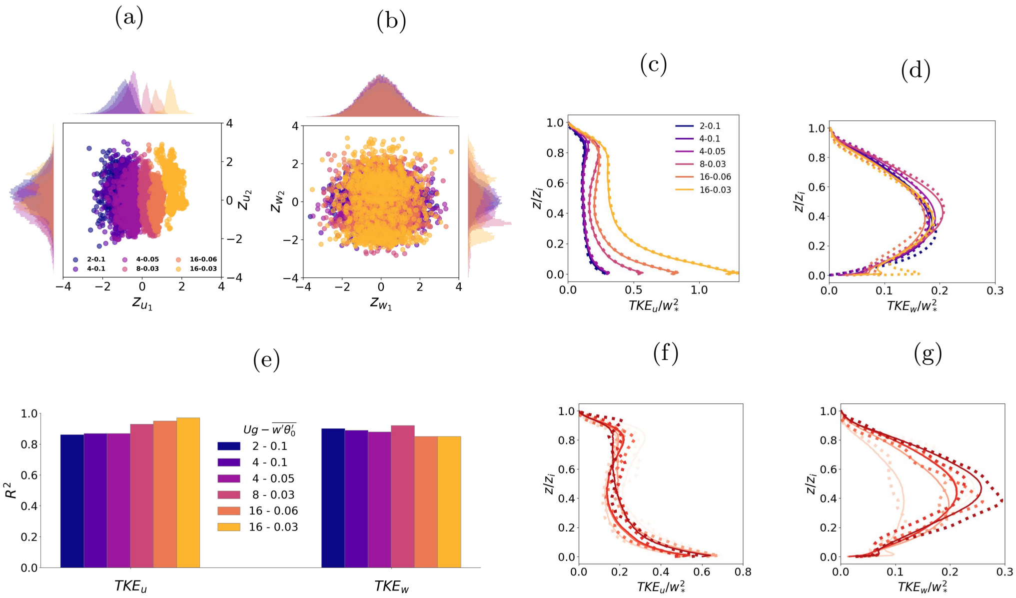


Figure 3: ED-TKE prediction: (a) displays the joint probability distribution of z_{u1} and z_{u2} extracted from the encoder trained on TKE profile. The marginal distributions are presented on the top (for z_{u1}) and the right side of the plot (for z_{u2}). (b) is similar to (a) but shows the joint probability distribution of z_w. Plot (c) displays the predicted (solid line) and true TKE_u (dashed line) averaged over each simulation, represented by colors. (d) is the same as (c) but for TKE_w. (e) shows the R^2 for TKE_u and TKE_w prediction. The colors represent different simulations, which are labeled in the legend as $Ug - \overline{w'\theta'_0}$. Finally, plots (f) and (g) respectively illustrate the networks' prediction (solid lines) and the true profiles (dashed lines) of TKE_u and TKE_w for randomly selected individual samples, distinguished by colors.

parabolic shape and primary peak. In simulations where the wind is strong (such as Ug16-$\overline{w'\theta'}_0$0.03 and Ug16-$\overline{w'\theta'}_0$0.06), a secondary peak in TKE_w near the surface is observed, which deviates slightly from the universal parabolic profile. However, our predicted TKE_w still exhibits this secondary peak, albeit with a smaller magnitude. The largest underestimation occurs for simulation Ug16-$\overline{w'\theta'}_0$0.03, where the predicted normalized secondary peak has a maximum of 0.1, while the true value is 0.18. We further emphasize that our networks are trained across regimes and are not targeting one specific regime, such as this mostly shear-driven mode.

To further investigate the ED-TKE skill in predicting TKE_u and TKE_w, we evaluate the predicted profiles for individual samples as shown in Figures 3f and 3g. These samples are randomly drawn from the test set. Although the mean profiles of TKE_u and TKE_w appear very smooth (Figures 3c and 3d), individual samples exhibit considerable variability (Figures 3f and 3g). The network captures the overall shape of each individual sample while smoothing out the small fluctuations observable in the true profiles. This behavior is consistent with existing literature (Takida et al., 2022) on the smoothness of encoder-decoder predictions and dimensionality reduction techniques. These methods only retain the information that is most relevant for the prediction, resulting in a smoother output. Also noted is that we did not include any information on horizontal neighboring cells in our network prediction, yet horizontal transport and variability in TKE, could lead to level-specific variations that cannot be captured by our strategy.

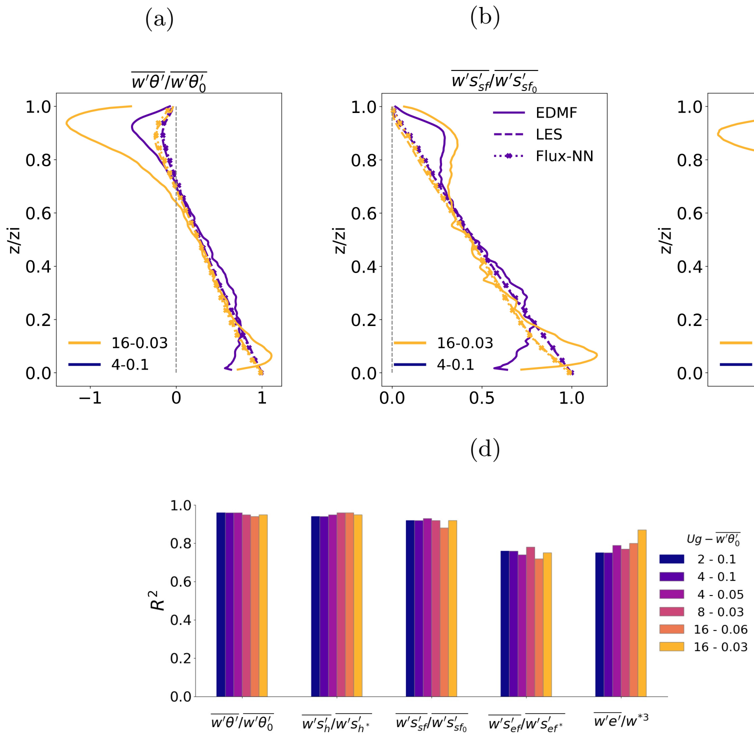


Figure 4: Plot shows the profiles of (a) vertical heat flux, (b) surface-forced tracer flux, and (c) entrainment-forced tracer flux, predicted by Flux-NN (dotted line), EDMF (solid lines), and computed from LES output (dashed line). Colors distinguish LES cases. Plot (d) shows R^2 computed for the neural network's prediction of turbulent fluxes for all simulations.

To quantify the skill of ED-TKE prediction, we compute R^2 for TKE_u and TKE_w and for each simulation separately (Figure 3e). R^2 is defined as one minus the ratio of the mean square error in prediction to the variance in the data. It ranges from zero to one, with one representing a perfect prediction with no error. For each simulation, we compute R^2 at each vertical level and then average layer-wise R^2 over all levels to obtain the final estimate (see Shamekh et al. (2022) for more detail). ED-TKE's prediction of TKE_u has a high R^2 (~ 0.9) across all simulations, while its prediction of TKE_w has a slightly lower R^2. Thus to summarize, our ED-TKE accurately captures relevant information for predicting TKE_u and TKE_w by only having access to TKE and shows a great performance across a large range of instability parameters present in the data set. We extract the latent variables from this network, z_u and z_v, to utilize as input for predicting vertical fluxes, as discussed in the next section.

4.2 Flux prediction

To predict the vertical turbulent fluxes of scalars and TKE, Flux-NN utilizes an encoder, e_x, to map the coarse-grained scalar or TKE profiles to a two-dimensional latent space (see Figure 1). These latent variables, along with z_u and z_w, are then processed by the decoders to predict the vertical turbulent flux profile of the corresponding scalar or TKE. In this section, we compare the Flux-NN predictions with fluxes directly computed from the coarse-grained LES output. Additionally, we compare our results with the reference ECMWF-implementation of EDMF scheme (Köhler et al., 2011), which has five tuning parameters. We re-tuned the EDMF parameters to obtain the best approximation of the heat flux for the Ug2-$\overline{w'\theta'}_0$0.1 run, which is most similar to the LES simulations utilized by Siebesma et al. (2007), in the originally developed parameterization. We subsequently use the re-tuned EDMF to predict the heat flux, surface-forced, and entrainment-forced tracer fluxes using their corresponding scalar profiles computed from our LES data (Figure 4).

The heat flux, normalized by its surface value, exhibits a universal profile as a function of normalized height z/z_i, decreasing linearly with height, reaching zero at the top of the mixed layer. In the inversion layer, the flux becomes negative and then approaches zero at the top of the boundary layer. Figure 4a illustrates the normalized turbulent heat fluxes predicted by Flux-NN (dotted lines), computed from LES outputs (dashed lines), and predicted by EDMF (solid lines) for two simulations one weakly and the other strongly unstable. The Flux-NN predictions closely match the coarse-grained fluxes computed from the LES for both illustrated cases (shear- or convectively-dominated) depicted in Figure 4 (and Figure S1). The EDMF scheme demonstrates reasonable heat flux prediction in the mixed layer, particularly for the strongly convective cases (as it was intended to). However, its prediction deviates from the LES output in the surface layer, exhibiting a considerable overestimation for the sheared cases (i.e., Ug16-$\overline{w'\theta'_0}$0.03). This overestimation decreases for cases with weak geostrophic wind, indicating the scheme's shortcomings in predicting fluxes for convective boundary layers with strong winds. Although we have discussed only two of the simulations for brevity, these findings are valid for our other simulations as well.

Remarkably, our Flux-NN accurately predicts the inversion layer heat flux across instability regimes (see Figure 4). The inversion layer flux presents a significant challenge for most traditional parameterizations, as it is strongly influenced by updrafts originating from the surface layer (Fedorovich et al., 2004), shear across the inversion (Pino et al., 2003, 2006; Pino & Vilà-Guerau De Arellano, 2008), and the entrainment of free tropospheric air into the boundary layer (Garcia & Mellado, 2014; Haghshenas & Mellado, 2019). Most traditional parameterizations do not explicitly incorporate the entrainment fluxes in their formulation and the entertainment is instead typically handled by the eddy-diffusion flux as in the EDMF, yet with important deviations. Indeed, as shown in Figure 4, the EDMF dramatically overestimates the magnitude of the heat flux in the inversion layer, particularly for the simulation with strong wind shear (e.i., Ug16-$\overline{w'\theta'}_0$0.03).

The Flux-NN is equally accurate in predicting the (normalized) surface-forced and entrainment-forced tracer fluxes, closely emulating the LES output (Figures 4b and 4c). This accuracy holds even in the inversion layer. However, EDMF significantly overestimates this part of the flux, particularly for entrainment-forced tracer, regardless of the geostrophic wind condition. This overestimation is related to the incorrect EDMF representation of the entrainment flux through the eddy diffusion. Given how important this entrainment is for key processes such as the diurnal growth of the PBL or shallow clouds formation and regimes, our new flux parameterization method might provide improvements to those key entrainment-related processes.

To further quantify the performance of Flux-NN, we computed the R^2 values separately for all simulations and fluxes (refer to Figure 4d). The R^2 values are very high (0.92-0.95) for $\overline{w'\theta'}$, $\overline{w's'_h}$, and $\overline{w's_{sf}}'$ across all simulations and turbulence regimes. However, for $\overline{w's_{ef}}'$ and $\overline{w'e'}$, the R^2 is smaller by about 0.1-0.15. Despite this, the flux prediction averaged over all samples of the same simulation is significantly close to the flux computed directly from the LES data for all scalars (Figure S1).

Additionally, to visualize the performance of Flux-NN at predicting individual samples, we randomly selected four samples for each scalar from the test data and plotted the predicted fluxes (solid lines) alongside the true fluxes (dashed lines) for these samples (Figure S1) with each sample distinguished by a different color. Despite the significant variability observed among samples of the same flux, particularly for $\overline{w's'_{ef}}$, $\overline{w's'_h}$, and $\overline{w'e'}$, Flux-NN accurately captures the overall shape of individual profiles while smoothing out fluctuations. This smoothing is similar to that observed in ED-TKE prediction and is related to the behavior of using reduced-order models, as discussed in section 4.1 and to the fact that we are not including the horizontal heterogeneity of the predictors in our vertical-only model. Thus, Flux-NN can predict vertical turbulent fluxes for various scalar profiles across a wide range of instability regimes, even in the inversion layer.

To summarize, Flux-NN accurately predicts turbulent fluxes of various scalars/TKE and provides a skillful approximation of all five fluxes across all six instability regimes (Figure 4d and Figure S1). Applying EDMF to the LES data reveals that this scheme does not generalize well to conditions with geostrophic winds or to tracers other than potential temperature. It overestimates the fluxes near the surface and in the inversion layer, particularly for entrainment-forced tracers, which rely heavily on the entrainment flux as the primary source of the scalar in the boundary layer. Additionally, the Flux-NN prediction of individual samples shows that the network can reproduce the overall shape of individual profiles while smoothing out fluctuations (Figure S1). This indicates that Flux-NN can predict the vertical turbulent fluxes of various scalars across a large range of instability regimes, even in the inversion layer. Therefore, it is a promising tool for modeling planetary boundary layers in climate and weather simulations.

4.3 Flux decomposition

The ED-TKE network discovers two separate latent variables that capture a hidden low-dimensional representation of horizontal and vertical TKE, which we refer to as z_u and z_w, respectively. The Flux-NN then utilizes these latent representations, along with z_x, to predict the contribution of each horizontal or vertical components to the total flux using Equation 12. We refer to each term in Equation 12 as a mode, with the first term ($\alpha_1 f_1(z_x, z_u)$) as the shear mode and the second term ($\alpha_2 f_2(z_x, z_w)$) as the convective mode. In this section, we discuss the shear and convective modes and their contributions to vertical turbulent fluxes, and investigate how this contribution changes across instability regimes. We primarily focus on turbulent heat, surface- and entrainment-forced tracer fluxes, while presenting results for TKE and height-dependent tracer fluxes in the supplementary material.

4.3.1 Vertical turbulent heat flux

Figure 5 illustrates the decomposition of the heat flux for all six simulations, with each mode normalized by its corresponding surface heat flux and plotted against the normalized height z/z_i. The shear mode (Figure 5b) is more prominent in simulations with a strong geostrophic wind, and its magnitude decreases as the instability parameter increases. In the most shear-driven simulation (e.g., Ug16-$\overline{w'\theta'}_0$0.03) the shear mode is responsible for approximately 80% of the total flux in the surface layer. Even in the mixed layer, the shear mode remains significant and explains about 70% of the flux. For the second most shear-driven simulation (e.g., Ug16-$\overline{w'\theta'}_0$0.06) and strongly convective cases, the contribution of the shear mode to the flux near the surface decreases from 75% and 50%, respectively. In these cases, the shear mode rapidly decreases with height, as expected, and becomes negligible in the mixed layer ($0.2 < z/z_i < 0.6$). In all simulations, the shear mode becomes negative in the upper part of the mixed layer ($z/z_i \sim 0.6-0.8$). In the inversion layer ($z/z_i \sim 0.8-1$), the shear mode increases (becomes more negative) with geostrophic wind, being more significant in highly sheared simulations.

Figure 5c depicts the convective modes of $\overline{w'\theta'}$ normalized by their respective surface heat flux and plotted as a function of z/z_i. The convective mode acts in the opposite direction to the shear mode and increases with instability, being larger for highly convective cases, as would be expected from basic understanding of the PBL. We note however that this behavior was not imposed but rather discovered by our networks when learning across simulation regimes. Despite differences in the instability parameters, the three most convective cases (Ug4-$\overline{w'\theta'}_0$0.05, Ug4-$\overline{w'\theta'}_0$0.1, and Ug2-$\overline{w'\theta'}_0$0.1) have very similar convective modes, which account for 50% of the flux near the surface and 100% in the mixed layer. Although one might expect the magnitude of the convective mode to increase with the PBL instability parameter, what we observe is that the convective mode is already quite large for Ug4-$\overline{w'\theta'}_0$0.05, which is in the free convective regime but has a smaller z_i/L compared to Ug4-$\overline{w'\theta'}_0$0.1 and Ug2-$\overline{w'\theta'}_0$0.1. Using quadrant analysis (Wyngaard & Moeng, 1992; D. Li & Bou-Zeid, 2011), Salesky et al. (2017) demonstrated that the heat transport efficiency also reaches a maximum past a given z_i/L threshold. Nonetheless, since their findings were based on quadrant analysis, we cannot make a direct comparison to our results.

In the inversion layer, the convective mode is strongest for simulations with larger instability parameters, thus Ug16-$\overline{w'\theta'}_0$0.03 and Ug16-$\overline{w'\theta'}_0$0.06 have the smallest contribution of convective mode into the flux in the inversion layer, and the three most unstable simulations have similar magnitudes.

The negative heat flux in the inversion layer has two sources: the overshoot of updrafts and the intrusion of free tropospheric air. The overshooting updrafts contain air with a negative θ anomaly and positive vertical velocity, thus creating a negative flux (Ghannam et al., 2017). On the other hand, the intrusion of free tropospheric air ventilates air with a positive θ anomaly and negative vertical velocity into the inversion layer, creating another negative heat flux. This intrusion is affected by the overshoot and wind shear in the inversion layer (Stull, 1976, 1973; Mcgrath-Spangler & Denning, 2010). Figure 5c suggests that the contribution of the convective mode to the inversion layer flux is larger for more convective cases, but it does not strongly scale with the surface heat flux or instability parameters. On the other hand, the intensity of the shear mode and its contribution to the inversion layer's flux depends on the strength of the wind shear. Thus, simulations Ug16-$\overline{w'\theta'}_0$0.03 and Ug16-$\overline{w'\theta'}_0$0.06 have the largest shear mode in the inversion layer. This finding is qualitatively consistent with that of Haghshenas and Mellado (2019); Garcia and Mellado (2014); Pino et al. (2003), showing the intensification of inversion layer flux with the wind shear.

4.3.2 Vertical turbulent surface-forced tracer flux

Figures 5e and 5f display the flux decomposition for the surface-forced tracer. The shear and convective modes of $\overline{w's_{sf}}$ highly resemble those of the turbulent heat flux, except in the inversion layer. The vertical flux of the surface-forced tracer is always positive, even in the inversion layer. This tracer has a source at the surface, and its concentration sharply decreases with height in the surface layer, then the tracer becomes nearly homogeneous vertically in the mixed layer (Figure 2). The surface-forced tracer concentration then rapidly decreases in the inversion layer, becoming zero in the free troposphere. The rising updrafts, which bring near-surface air with positive tracer anomaly into the inversion layer, create a positive flux. On the other hand, the entrainment flux injects free tropospheric air with a negative velocity and negative tracer anomaly (as they have a value of exactly zero above) into the inversion layer, generating a positive flux. Thus, the reduction of the surface-forced tracer concentration in the inversion layer results in its flux having the opposite sign of the heat flux one (Figure 5).

4.3.3 Vertical turbulent entrainment-forced tracer flux

Figure 2 shows the entrainment-forced tracer profile and its corresponding vertical turbulent flux computed from LES data, and Figure 5g shows the predicted flux for all simulations. Additionally, Figure 4c compares the predicted flux with the flux cal-

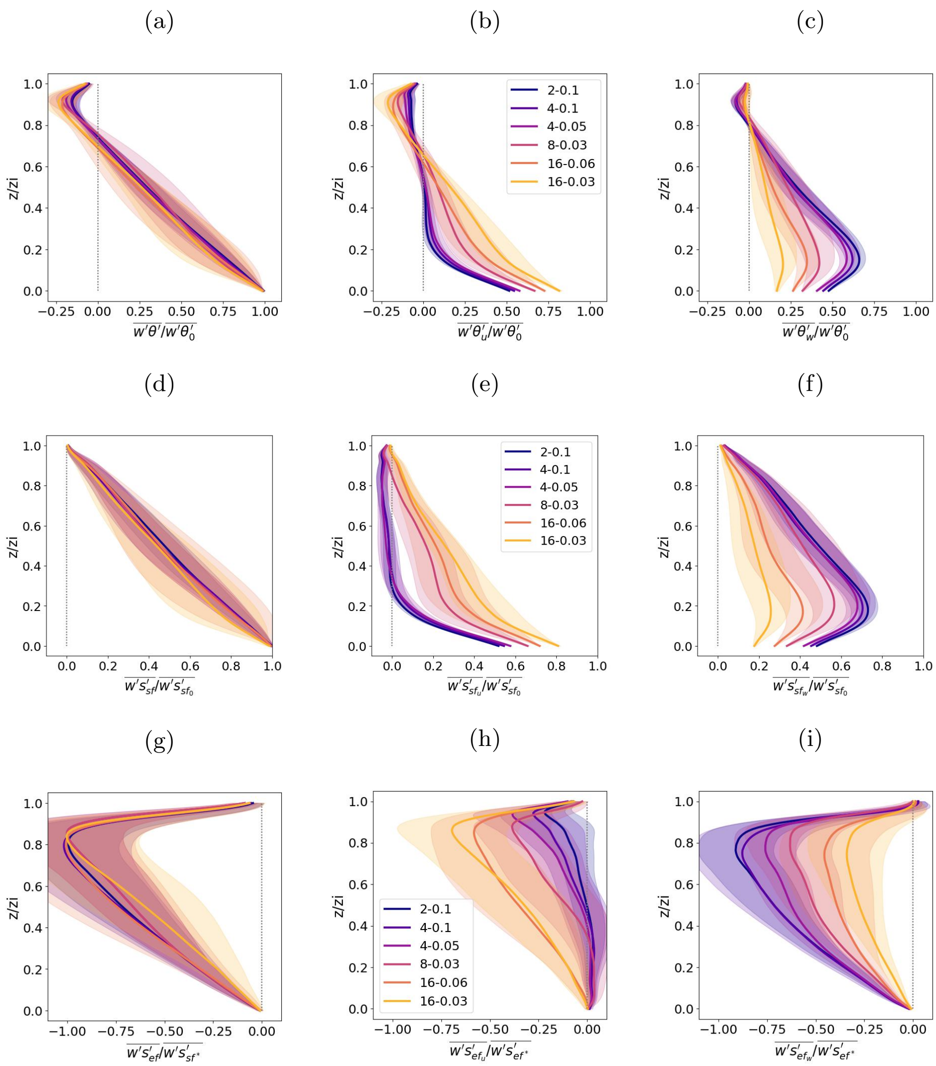


Figure 5: Plot shows (a) the vertical turbulent heat flux for various simulations, (b) shear mode represented as $\alpha_1 f_1$ in Equation 12.a, (b) convective mode represented as $\alpha_2 f_2$ in Equation 12.b, for heat flux decomposition. Plots d-f and g-i show the same as a-c but for surface-forced, and entrainment-forced tracer flux, respectively. The colors represent different simulations, which are labeled in the legend as $Ug - \overline{w'\theta'_0}$.

culated from LES data. This flux is negative across all six simulations. Figures 5h and 5i display shear and convective modes of the flux predicted by flux-NN. The shear mode of the strongly convective simulations is nearly zero from the surface to the middle of the mixed layer, at $z/z_i \sim 0.5$, indicating that the convective mode is mostly responsible for the flux at these layers. The significant contribution of the convective mode to the total flux highlights the importance of convective transport for the entrainment-forced tracer, despite the absence of a source near the surface or within the PBL. The only source of this tracer is the ventilation of free tropospheric air with a high tracer concentration into the boundary layer. Thus, the entrainment flux and downdraft play an essential role in this flux, bringing air with high tracer concentration downward, causing a negative flux. However, the updraft also contributes greatly to this flux by transporting near-surface air with a low tracer concentration upward, resulting in a negative flux. The role of the updraft in generating a vertical turbulent flux of entrainment-forced tracer, also known as top-down tracer, is often overlooked (Chor et al., 2020; Wyngaard & Brost, 1984). This is likely because the flux of this tracer can be fully explained by eddy-diffusivity models by assigning a large enough eddy diffusivity, as the flux is always down concentration gradient. Thus, since this tracer has no source near the surface, the role of updrafts in its flux is often disregarded (Chor et al., 2020). We show here that this is not the case. Our quadrant and subdomain-division analysis provide further confirmation of the significant contribution of updrafts and non-diffusive transport to the vertical turbulent flux of the entrainment-forced tracer (not shown).

In this section, we have discussed our approach of using a range of turbulent regimes, from shear-dominant to convective-dominant, to develop a constraint that enables us to decompose the total flux into two modes of variability. While there is no ground truth to accurately quantify our flux decomposition, we can qualitatively evaluate the two modes based on our physical understanding of turbulent flow and how the forcing can affect the flow. We also examined the flux decomposition for heat, surface- and entrainment-forced tracers and discussed the role of convective and shear modes in the vertical turbulent flux. Overall, the flux decomposition approach provides insight into the underlying mechanisms of turbulent flow and can be used to better understand and model the boundary layer dynamics.

4.4 Mode-specific estimation of diffusive flux using neural network

As mentioned in the introduction, most parameterizations of turbulent flux decompose the vertical turbulent flux into a diffusion and a non-diffusion term. Typically, the eddy diffusivity K needs to be parameterized, but there is no unique approach for doing so. Holtslag and Moeng (1991) define an eddy diffusivity using a simplified turbulent heat flux equation. This eddy diffusivity, which is related to the variance of vertical velocity, is adapted by Siebesma et al. (2007) for their EDMF scheme. Chor et al. (2020) estimate the diffusive and non-diffusive flux by maximizing for the diffusive part. Q. Li et al. (2021) employ a sub-domain decomposition approach and Taylor series expansion of the updraft and downdraft mass-flux transport to approximate down-gradient flux and then the eddy diffusivity. Lopez-Gomez et al. (2020) define an eddy mixing length based on constraints derived from the TKE balance.

While our TKE-based decomposition does not enforce a flux separation based on methods such as eddy length-scale or diffusivity, we are still interested in understanding the extent to which our extracted shear- and convective-modes exhibit diffusive behavior. To investigate this, we project each mode onto the vertical gradient of its corresponding scalar and determine the contribution of its diffusive part by maximizing the linear profile to the total flux. We use a regression neural network to predict an eddy diffusivity and compute the diffusive flux using Equation 1. As Figure B1 shows, for each vertical layer of the PBL, we calculate the vertical gradient of the scalars. Then, we input the TKE and the distance from the surface, z/z_i, of that layer into a neural network which outputs an eddy diffusivity value (K) for that specific layer. Next, we multiply K by the local gradient of the scalar (as per Equation 1) to estimate the total diffusive flux at that particular level. Although we do not have access to any ground truth value for the diffusive flux to use as a target value for supervised learning, we train the neural network to maximize the contribution of the diffusive flux to the total flux. In other words, we use our two modes f_1 and f_2 as the target value so that the network can predict an eddy diffusion flux that best matches these modes. Chor et al. (2020) used a similar approach to decompose the total flux into diffusive and non-diffusive components, but they predicted the entire vertical turbulent flux, whereas in our study, we project on each mode separately. This means that we determine the diffusive part of each mode, resulting in two eddy diffusivities, K_u and K_w, representing the eddy diffusivities of the shear and convective modes, respectively. We assume that these two K values are the

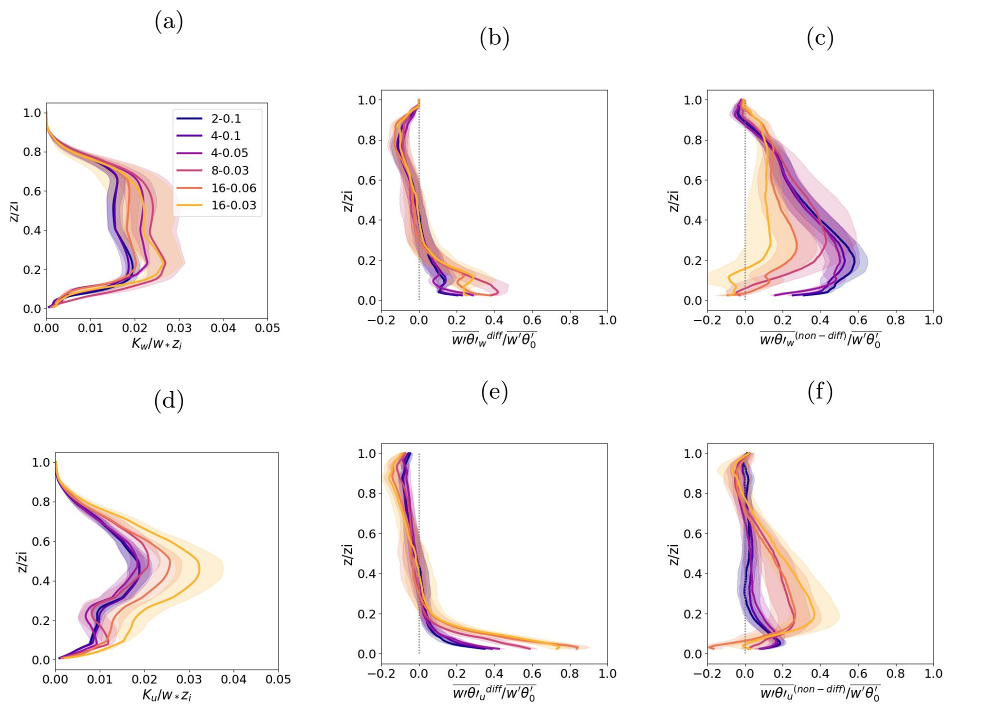


Figure 6: The plots depict the diffusive component of each mode of the vertical turbulent heat flux. In plot (a), the eddy diffusivity of the convective mode, denoted as K_w, is computed using a neural network. Plot (b) illustrates the diffusive portion of convective mode, while plot (c) shows the non-diffusive portion of the convective mode of the heat flux. Similarly, plots (d) to (f) present the corresponding information for the shear mode. The colors represent different simulations, which are labeled in the legend as $Ug - \overline{w'\theta'_0}$.

same for all scalars within the same simulation but vary across simulations. This assumption naturally constrains K_u and K_w, and we can express this projection as:

.

$$\overline{w'x'(z)}_w^{diff} = -NN_w(TKE_w(z), z/z_i).(\frac{\partial \overline{X}}{\partial z}(z)) \quad (14)$$

$$\overline{w'x'(z)}_u^{diff} = -NN_u(TKE_u(z), z/z_i).(\frac{\partial \overline{X}}{\partial z}(z)) \quad (15)$$

We use the neural network NN_w to predict the eddy diffusivity K_w and NN_u to predict K_u. After training the network and approximating the diffusive flux, we calculate the non-diffusive flux as a residual:

$$\overline{w'x'}_u^{Non-Diff} \sim \overline{w'x'}_u - (-K_u \frac{\partial \overline{X}}{\partial z})$$

for the shear mode and:

$$\overline{w'x'}_w^{Non-Diff} \sim \overline{w'x'}_w - (-K_w \frac{\partial \overline{X}}{\partial z})$$

for the convective mode. A detailed explanation of the neural network, its loss function, and the projection is provided in the Appendix B.

Figure 6a and 6d display the eddy diffusivity K_u and K_w normalized by $w_* z_i$, respectively, and plotted versus the normalized height z/z_i. To facilitate comparison with previously suggested eddy diffusivity, we plotted the eddy diffusivity computed based on Holtslag and Moeng (1991), hereafter K_H, shown in black lines in Figures S3, as a reference.

In Figure 6b and 6e, we present the diffusive parts of shear and convective mode, computed for the heat flux. The diffusive shear mode is significant in the surface layer but quickly diminishes to zero at approximately $z/z_i > 0.2$, and remains close to zero for $0.2 < z/z_i < 0.6$, where the vertical potential temperature gradient is insignificant. Therefore, a substantial portion of the shear mode, even for weakly convective cases, is non-diffusive (Figure 6f).

In the upper part of the mixed layer ($z/z_i > 0.6$), the diffusive shear flux becomes negative for both shear-driven and convective-driven cases. Interestingly, in the inversion layer, the shear mode is composed of both diffusive and non-diffusive components in shear-driven cases, but only the diffusive component is present in convective-driven cases. Similar to the shear mode, the convective mode (Figure 6b-c) is mostly non-diffusive except in the surface and inversion layers. In the inversion layer the diffusive convective mode is negative for all cases, and explains all convective mode flux.

Overall, we find that the two modes learned by the neural network are mostly non-diffusive, except in the surface and inversion layer. Additionally, the eddy diffusivity that we learn is about three times smaller than the eddy diffusivity suggested by Holtslag and Moeng (1991), as shown in Figure S3. The small magnitude of the diffusive flux implies that the Flux-NN model does not heavily rely on the diffusion term to predict the shear and convective modes. The model's latent variables can capture complex structures and learn both linear and non-linear relationships between scalars and fluxes, rather than just down-gradient ones.

Furthermore, when projecting the modes onto the scalar gradients, the neural network must simultaneously provide a down-gradient diffusive flux for all scalars, which places a stronger constraint on the magnitude of K. In other words, the diffusive flux must be down-gradient for all scalars, and learning an eddy diffusivity for only one scalar does not guarantee a down-gradient flux for a different scalar. Conventional parameterization often learns an eddy diffusivity term that compensates for neglected processes, such as down-draft or entrainment, resulting in an unrealistically large eddy diffusivity. This approach is commonly used in ocean mixed layer modeling.

5 Discussion and conclusion

To predict turbulent transport in the planetary boundary layer in numerical weather prediction and climate models, parameterizations have been widely adopted due to the models' limited spatial resolution. Historically, various approaches have been employed to parameterize turbulence, primarily based on scale separation, where separate schemes have been developed to represent small scale eddies and large scale coherent structures. In this work we focus on the dry convective boundary layer under different regimes from shear- to convective-dominated regimes and employ machine learning tools to develop a data-driven parameterization of vertical turbulent fluxes of various scalars and across a large range of instability regimes.

Although machine learning has become a popular tool for emulating physical processes, it faces two major issues: its high dimensionality that limits physical interpretability and therefore trust, and it typically lacks the integration of physical constraints into its emulators. In this work, we take a significant step towards solving these issues by introducing a lower-dimensional, latent representation of turbulent transport in the planetary boundary layer by introducing a physical constraint that enables us to decompose the flux into two main modes of variability. Our findings demonstrate that the latent representation of turbulent kinetic energy (TKE) can encode information related to the vertical and horizontal components of TKE, which reflect the relative contributions of thermal and mechanical turbulence to the vertical turbulent flux of a scalar. This is consistent with the fact that the turbulent flux in the boundary layer is primarily generated by the mechanical and buoyancy effects of wind shear and convection interacting with the flow, respectively. To ensure a separate representation of vertical and horizontal TKE in the latent space of TKE, we applied a physical constraint through the architecture of our neural network. Our approach involves using an encoder-decoder network that takes total TKE as input, which is readily available in most boundary layer parameterizations. By encapsulating the essential structural information needed for separately predicting horizontal and vertical TKE when given only total TKE as input, our network can effectively capture the relevant information for predicting these components. The TKE latent representation is then used to predict the vertical turbulent fluxes.

We showed that by reducing the dimension of TKE into two latent representations corresponding separately to horizontal and vertical TKE, we can accurately decompose the vertical flux of any scalar into two modes using a second set of neural networks. One of these modes is associated with horizontal TKE, which we refer to as a shear-driven mode, while the second mode is associated with vertical TKE and is called the convective mode. This flux decomposition is distinct from traditional schemes because it enables us to learn how each forcing contributes to the total flux and quantify their fractional contribution. By training the neural network on a wide range of scalars and simulations, we enable it to approximate a unique function for each mode that is independent of the scalar profile and turbulent regime. Additionally, these two modes and their variations with instability parameters are qualitatively consistent with our understanding of convection and shear contribution to the boundary layer vertical turbulent fluxes at various instability parameters.

Our analysis helps further refine our understanding of turbulent transport in the boundary layer and reveals that the neural network does not rely on the local gradient to generate the vertical turbulent fluxes. Specifically, by projecting each mode onto the gradient of its corresponding scalar, we observe that the fluxes are mostly non-diffusive, except in the surface and inversion layers. Even for entrainment-forced tracers, which exhibit fluxes down the gradient, the fluxes appear to be non-diffusive in our approach. In contrast, Chor et al. (2020) found that entrainment-forced tracer fluxes can be explained through diffusive fluxes even for the most convective case they studied. The contrasting results may stem from our neural network, which decomposes the flux without enforcing the gradient-following behavior, as opposed to their conventional diffusive approach. Our approach provides a unified framework to learn how each forcing contributes to the flux, offering insights into the underlying physical processes of turbulence in boundary layers.

We trained our neural network on a series of simulations, with instability parameters ranging from weakly unstable to strongly unstable. Our tests on the generalization of this network to unseen instability parameters indicate that the network exhibits skillful performance in interpolation. Specifically, when a simulation with an instability parameter between the minimum and maximum instability parameters present in the dataset is removed from the training set and used as a test set, the resulting R^2 value exceeds 0.8. Moreover, the network shows reasonable extrapolation capabilities when tested on cases with instability parameters larger than the range of instability parameters used in the training set. For example, when we remove the most convective simulation (Ug2-$\overline{w'\theta'}_0$0.1) from the training set and use it as a test set, the resulting R^2 value equals 0.75.

Hence, the model effectively extrapolates to unseen purely convective cases. This may be due to the fact that the non-dimensionalized profiles of TKE and scalars become similar at high instability parameters.

However, the network exhibits limitations in extrapolating to cases where the instability parameter is smaller than that of the training set. Removing the most shear driven simulation (Ug16-$\overline{w'\theta'}_0$0.03) from the training set and using it as a test set results in an R^2 value of 0.5. We attribute this shortcoming to the dynamics of the boundary layer turbulence, which become markedly different when the system approaches the neutral situation. Additionally, the non-dimensionalized fluxes and TKE profiles exhibit self-similarity for unstable simulations, leading to great extrapolation performance for both ED-TKE and flux-NN. However, for simulations with smaller instability parameters (i.e., near neutral turbulent regime), the non-dimensionalization does not result in a self-similar profile, making the extrapolation to simulations with instability parameters smaller than those in the training data much more challenging. In conventional parameterization of climate models, the three cases of stable, neutral, and convective conditions are often treated using three (or, in some cases, two) separate schemes, by switching from one scheme to another at a certain instability parameter which is, itself, set arbitrarily. This caveat is the subject of our future research to develop a parameterization that accurately models across a large range of instability parameters from strongly stable to strongly unstable situations.

One limitation of this study is the scale and grid dependency of our data-driven parameterization. Specifically, we coarse-grain the LES data to grids of 1.5x1.5 km^2, which lies within the "gray zone" of grid scales. Coarse-graining the data to a different grid size would alter the coarse profile of scalars and TKE, rendering the neural network trained on the original coarse data inaccurate for modeling other coarse data beyond the training set. In other words, our parameterization is not yet scale-adaptive. Furthermore, our network is trained on a specific vertical grid spacing and is, thus, sensitive to the grid spacing of the test data. Ideally, we aim to develop a model that is grid-agnostic such that it can be easily integrated into any weather or climate model, regardless of the horizontal grid size and vertical gird spacing used in the original data. We recognize this shortcoming and plan to address it in future research.

Appendix A Loss function

Variational Autoencoders (VAEs) take a Bayesian perspective and assume that the input to the encoder is generated from a conditional probability distribution that describes an underlying generative model. The multivariate latent representation of the input, denoted as z, is assumed to follow a prior distribution P(z). The model is then trained to maximize the probability of generating samples in the training dataset by optimizing both the reconstruction loss and the Kullback-Leibler divergence (KL divergence) of the approximate posterior, which is assumed to be Gaussian, from the prior distribution. Instead of predicting a single n-dimensional latent representation, the encoder predicts a mean and a standard deviation. The KL divergence term forces this distribution to be close to the prior distribution, which is typically assumed to be a normal distribution. This helps to enforce a disentanglement in the latent variables learned by the encoder, which is a property of interest in our work. Additionally, predicting a distribution instead of a single value results in a continuous latent space, which is valuable for using our neural network as a generator for parameterization. Therefore, we include the KL divergence in our loss.

We employ a variational encoder-decoder architecture, where we approximate the underlying generative model but instead of reconstructing the input TKE, we predict the horizontal and vertical TKE. Hence, our approach involves supervised training rather than unsupervised training. The loss consists of four terms: two are the mean squared errors of the predictions, and the other two are the KL divergences of the latent representations of the horizontal and vertical TKE.

The loss of predicting horizontal and vertical TKE is:

$$L_{MSE} = \frac{1}{N}\left(\sum_{i=1}^{N}\sum_{j=1}^{D}(TKE_u^t - TKE_u^p)_{ij}^2 + \sum_{i=1}^{N}\sum_{j=1}^{D}(TKE_w^t - TKE_w^p)_{ij}^2\right) \tag{A1}$$

where t represents the ground-truth coarse-grained profiles computed directly from LES, and p represents the coarse-grain profiles predicted by neural network. N represents the batch size and D is the dimension of the input which is 128.

The KL divergence loss, given the assumption of normal distribution for prior, is as follow

$$L_{KL_D} = \frac{1}{N} * \frac{1}{d}\left(\sum_{i=1}^{N}\sum_{k=1}^{d}(1 - ln\sigma_{u_{ik}}^2 + \mu_{u_{ik}}^2 + \sigma_{u_{ik}}^2) + \sum_{i=1}^{N}\sum_{k=1}^{d}(1 - ln\sigma_{w_{ik}}^2 + \mu_{w_{ik}}^2 + \sigma_{w_{ik}}^2)\right) \tag{A2}$$

where μ is the mean and σ is the standard deviation predicted by the encoder. d is the dimension of latent space, here equal to two and N is the batch size.

The total loss of ED-TKE is then the sum of the two terms:

$$loss_{ED} = L_{MSE} + \lambda L_{KL_D} \tag{A3}$$

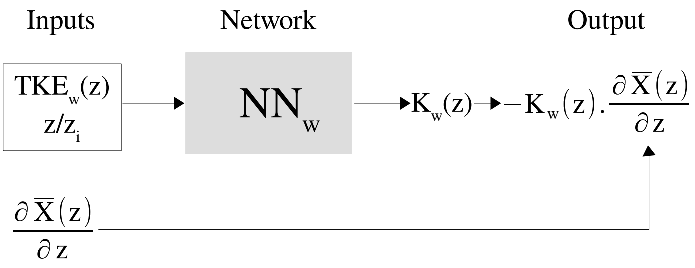
λ is a hyperparameter that we empirically set to 10^{-1}. Assigning a lager value to λ increases the reconstruction error while assigning a smaller value reduces the Gaussianization of the distribution of the latent variables and their disentanglement. Gaussianization and disentanglement are desirable because many statistical models assume that the data is normally distributed, and by transforming the data to be closer to a Gaussian distribution, it can be easier to model and analyze the data. In the context of deep learning, Gaussianization can also help to regularize the learning process and prevent overfitting. Disentanglement refers to the property of the latent space where each dimension of the space represents a distinct and independent factor of variation in the data. This means that different aspects of the data are represented by different dimensions in the latent space, allowing for more precise manipulation and control of the data. Disentanglement can also help with interpretability and understanding of the model, as it provides a clear mapping between the latent space and the original data space. Therefore, by promoting Gaussianization and disentanglement in the latent space, we can improve the interpretability, flexibility, and generalization performance of the model.

The loss of Flux-NN is constructed the same way, by combining the KL divergence term with the MSE of flux prediction. This loss is then:

$$loss_{flux} = \frac{1}{N} * \frac{1}{D}\sum_{i=1}^{N}\sum_{j=1}^{D}(\overline{w'x'}_{ij}^{t} - \overline{w'x'}_{ij}^{p})^2 + \frac{1}{N} * \frac{1}{d}\sum_{i=1}^{N}\sum_{k=1}^{d}(1 - ln\sigma_{x_{ik}}^2 + \mu_{x_{ik}}^2 + \sigma_{x_{ik}}^2) \tag{A4}$$

Appendix B Predicting diffusive flux

Section 4.4 employs a neural network to predict the eddy diffusivity and, consequently, the diffusive component of each mode of variability of the turbulent fluxes. This appendix provides additional details on the network's architecture and its training process. Figure B1 displays the network's architecture and its associated loss function. The neural network takes layer-wise TKE and z/zi as inputs and generates a predicted value



$$\text{loss} = \left(\overline{w'x'}_w(z) - \left(-K_w(z) . \frac{\partial \overline{X}(z)}{\partial z}\right)\right)^2$$

Figure B1: The neural network uses inputs such as $TKE_w(z)$ ($TKE_u(z)$) and z/z_i, representing the distance to the surface, to predict the eddy diffusivity $K_w(z)$ ($K_u(z)$). This eddy diffusivity is then multiplied by the scalar gradient to generate the output, which represents the diffusive flux. The network is trained with the target value of the convective (shear) mode, which compels the model to predict a diffusive flux as close as possible to the convective (shear) mode.

for eddy diffusivity. This predicted value is then multiplied by the gradient of the scalar, such as $\frac{\partial \overline{\theta}(z/zi)}{\partial z}$, resulting in the final prediction of the neural network. The network utilizes the convective (shear) mode as its target, meaning that it attempts to maximize the predicted diffusive component of each mode. This approach is similar to the one employed by (Chor et al., 2020), except that they did not use a neural network for their optimization.

The fully connected feed-forward neural network used in this study consists of four layers with 32, 64, 32, and 8 neurons in each layer, respectively. The final layer of the network, responsible for outputting the eddy diffusivity, employs a rectified linear unit (ReLU) activation function to ensure that the predicted eddy diffusivity remains positive. The network is trained using a batch size of 512 for 50 epochs, employing early stopping with a patience of five.

Open Research Section

The machine learning tools developed for this study as well as the scripts for pre- and post-processing data can be found here: `https://doi.org/10.5281/zenodo.8039033`

Acknowledgments

SS and PG acknowledge funding from European Research council grant USMILE, from Schmidt Future project M2LiNES and from the National Science Foundation Science and Technology Center (STC) Learning the Earth with Artificial intelligence and Physics (LEAP), Award # 2019625 - STC. SS is particularly grateful to Qi Li and Yuanfeng Cui for their generous assistance with large eddy simulations.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... others (2016). Tensorflow: a system for large-scale machine learning. In *Osdi* (Vol. 16, pp.

265–283).
Albertson, J. D. (1996). *Large eddy simulation of land-atmosphere interaction.* University of California, Davis.
Albertson, J. D., & Parlange, M. B. (1999). Surface length scales and shear stress: Implications for land-atmosphere interaction over complex terrain. *Water Resources Research*, *35*(7), 2121-2132. doi: https://doi.org/10.1029/1999WR900094
Angevine, W. M., White, A. B., & Avery, S. K. (1994). Boundary-layer depth and entrainment zone characterization with a boundary-layer profiler. *Boundary-Layer Meteorology*, *68*(4), 375–385.
Bardina, J., Ferziger, J., & Reynolds, W. (1980). Improved subgrid-scale models for large-eddy simulation. In *13th fluid and plasmadynamics conference* (p. 1357).
Behrens, G., Beucler, T., Gentine, P., Iglesias-Suarez, F., Pritchard, M., & Eyring, V. (2022). Non-linear dimensionality reduction with a variational encoder decoder to understand convective processes in climate models. *Journal of Advances in Modeling Earth Systems*, *14*(8), e2022MS003130. (e2022MS003130 2022MS003130) doi: https://doi.org/10.1029/2022MS003130
Betts, A. (1973). Non-precipitating cumulus convection and its parameterization. *Quarterly Journal of the Royal Meteorological Society*, *99*(419), 178–196.
Beucler, T., Pritchard, M. S., Yuval, J., Gupta, A., Peng, L., Rasp, S., ... Gentine, P. (2021). Climate-invariant machine learning. *CoRR*, *abs/2112.08440*.
Bolton, T., & Zanna, L. (2019). Applications of deep learning to ocean data inference and subgrid parameterization. *Journal of Advances in Modeling Earth Systems*, *11*(1), 376-399. doi: https://doi.org/10.1029/2018MS001472
Bou-Zeid, E., Meneveau, C., & Parlange, M. (2005). A scale-dependent lagrangian dynamic model for large eddy simulation of complex turbulent flows. *Physics of Fluids*, *17*(2), 025105. doi: 10.1063/1.1839152
Caruana, R., Lawrence, S., & Giles, C. (2000). Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. *Advances in neural information processing systems*, *13*.
Chen, R. T., Li, X., Grosse, R. B., & Duvenaud, D. K. (2018). Isolating sources of disentanglement in variational autoencoders. *Advances in neural information processing systems*, *31*.
Cheng, Y., Giometto, M., Kauffmann, P., Lin, L., Cao, C., Zupnick, C., ... Gentine, P. (2019). Deep learning for subgrid-scale turbulence modeling in large-eddy simulations of the atmospheric boundary layer. *arXiv preprint arXiv:1910.12125*.
Chinita, M. J., Matheou, G., & Teixeira, J. (2018). A joint probability density–based decomposition of turbulence in the atmospheric boundary layer. *Monthly Weather Review*, *146*(2), 503 - 523. doi: 10.1175/MWR-D-17-0166.1
Chor, T., McWilliams, J. C., & Chamecki, M. (2020). Diffusive–nondiffusive flux decompositions in atmospheric boundary layers. *Journal of the Atmospheric Sciences*, *77*(10), 3479–3494.
Corrsin, S. (1975). Limitations of gradient transport models in random walks and in turbulence. In F. Frenkiel & R. Munn (Eds.), *Turbulent diffusion in environmental pollution* (Vol. 18, p. 25-60). Elsevier. doi: https://doi.org/10.1016/S0065-2687(08)60451-3
Deardorff, J. (1972). Theoretical expression for the countergradient vertical heat flux. *Journal of Geophysical Research*, *77*(30), 5900–5904.
Deardorff, J. W. (1974). Three-dimensional numerical study of turbulence in an entraining mixed layer. *Bound.-Layer Meteor.*, *7*, 199–226.
Deardorff, J. W., et al. (1970). Convective velocity and temperature scales for the unstable planetary boundary layer and for rayleigh convection. *J. atmos. Sci*, *27*(8), 1211–1213.

Doersch, C. (2016). Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*.

Ertel, H. (1942). Der vertikale turbulenz-wärmestrom in der atmosphäre. *Meteor. Z*, *59*, 250–253.

Fedorovich, E., Conzemius, R., & Mironov, D. (2004). Convective entrainment into a shear-free, linearly stratified atmosphere: Bulk models reevaluated through large eddy simulations. *Journal of the atmospheric sciences*, *61*(3), 281–295.

Garcia, J. R., & Mellado, J. P. (2014). The two-layer structure of the entrainment zone in the convective boundary layer. *Journal of the Atmospheric Sciences*, *71*(6), 1935–1955.

Gentine, P., Bellon, G., & van Heerwaarden, C. C. (2015). A closer look at boundary layer inversion in large-eddy simulations and bulk models: Buoyancy-driven case. *Journal of the atmospheric Sciences*, *72*(2), 728–749.

Gentine, P., Pritchard, M., Rasp, S., Reinaudi, G., & Yacalis, G. (2018). Could machine learning break the convection parameterization deadlock? *Geophysical Research Letters*, *45*(11), 5742–5751.

Ghannam, K., Duman, T., Salesky, S. T., Chamecki, M., & Katul, G. (2017). The non-local character of turbulence asymmetry in the convective atmospheric boundary layer. *Quarterly Journal of the Royal Meteorological Society*, *143*(702), 494-507. doi: https://doi.org/10.1002/qj.2937

Haghshenas, A., & Mellado, J. P. (2019). Characterization of wind-shear effects on entrainment in a convective boundary layer. *Journal of Fluid Mechanics*, *858*, 145–183. doi: 10.1017/jfm.2018.761

Han, J., Witek, M. L., Teixeira, J., Sun, R., Pan, H.-L., Fletcher, J. K., & Bretherton, C. S. (2016). Implementation in the ncep gfs of a hybrid eddy-diffusivity mass-flux (edmf) boundary layer parameterization with dissipative heating and modified stable boundary layer mixing. *Weather and Forecasting*, *31*(1), 341 - 352. doi: 10.1175/WAF-D-15-0053.1

Hertel, L., Collado, J., Sadowski, P., Ott, J., & Baldi, P. (2020). Sherpa: Robust hyperparameter optimization for machine learning. *SoftwareX*. (Also arXiv:2005.04048. Software available at: https://github.com/sherpa-ai/sherpa)

Holtslag, A., & Moeng, C.-H. (1991). Eddy diffusivity and countergradient transport in the convective atmospheric boundary layer. *Journal of the Atmospheric Sciences*, *48*(14), 1690–1698.

Kalina, E. A., Biswas, M. K., Zhang, J. A., & Newman, K. M. (2021). Sensitivity of an idealized tropical cyclone to the configuration of the global forecast system–eddy diffusivity mass flux planetary boundary layer scheme. *Atmosphere*, *12*(2). doi: 10.3390/atmos12020284

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P., & Welling, M. (2022). *Auto-encoding variational bayes.*

Köhler, M., Ahlgrimm, M., & Beljaars, A. (2011). Unified treatment of dry convective and stratocumulus-topped boundary layers in the ecmwf model. *Quarterly Journal of the Royal Meteorological Society*, *137*(654), 43–57.

Kumar, R., & Adrian, R. J. (1986). Higher order moments in the entrainment zone of turbulent penetrative thermal convection. *J. Heat Transfer*, *108*, 323–329.

Kumar, V., Kleissl, J., Meneveau, C., & Parlange, M. B. (2006). Large-eddy simulation of a diurnal cycle of the atmospheric boundary layer: Atmospheric stability and scaling issues. *Water Resources Research*, *42*(6). doi: https://doi.org/10.1029/2005WR004651

Lenschow, D. H., Lothon, M., Mayor, S. D., Sullivan, P. P., & Canut, G. (2012). A comparison of higher-order vertical velocity moments in the convective boundary layer from lidar with in situ measurements and large-eddy simulation. *Boundary-layer meteorology*, *143*, 107–123.

Lenschow, D. H., Wyngaard, J. C., & Pennell, W. T. (1980). Mean-field and second-moment budgets in a baroclinic, convective boundary layer. *Journal of Atmospheric Sciences*, *37*(6), 1313 - 1326. doi: https://doi.org/10.1175/1520-0469(1980)037⟨1313:MFASMB⟩2.0.CO;2

Li, D., & Bou-Zeid, E. (2011). Coherent structures and the dissimilarity of turbulent transport of momentum and scalars in the unstable atmospheric surface layer. *Boundary-Layer Meteorology*, *140*, 243–262.

Li, Q., Cheng, Y., & Gentine, P. (2021). Connection between mass flux transport and eddy diffusivity in convective atmospheric boundary layers. *Geophysical Research Letters*, *48*(8), e2020GL092073.

Li, Q., Gentine, P., Mellado, J. P., & McColl, K. A. (2018). Implications of nonlocal transport and conditionally averaged statistics on monin–obukhov similarity theory and townsend's attached eddy hypothesis. *Journal of the Atmospheric Sciences*, *75*(10), 3403–3431.

Lopez-Gomez, I., Cohen, Y., He, J., Jaruga, A., & Schneider, T. (2020). A generalized mixing length closure for eddy-diffusivity mass-flux schemes of turbulence and convection. *Journal of Advances in Modeling Earth Systems*, *12*(11), e2020MS002161. (e2020MS002161 10.1029/2020MS002161) doi: https://doi.org/10.1029/2020MS002161

Ma, X., Lin, Y., Nie, Z., & Ma, H. (2020). Structural damage identification based on unsupervised feature-extraction via variational auto-encoder. *Measurement*, *160*, 107811.

Mathieu, E., Rainforth, T., Siddharth, N., & Teh, Y. W. (2019). Disentangling disentanglement in variational autoencoders. In *International conference on machine learning* (pp. 4402–4412).

Mcgrath-Spangler, E., & Denning, S. (2010). Impact of entrainment from overshooting thermals on land–atmosphere interactions during summer 1999. *Tellus B: Chemical and Physical Meteorology*, *62*(5), 441–454.

Meng, Q., Catchpoole, D., Skillicom, D., & Kennedy, P. J. (2017). Relational autoencoder for feature extraction. In *2017 international joint conference on neural networks (ijcnn)* (pp. 364–371).

Moeng, C.-H., & Wyngaard, J. C. (1984). Statistics of conservative scalars in the convective boundary layer. *Journal of Atmospheric Sciences*, *41*(21), 3161–3169.

Moeng, C.-H., & Wyngaard, J. C. (1989). Evaluation of turbulent transport and dissipation closures in second-order modeling. *Journal of the Atmospheric Sciences*, *46*(14), 2311–2330.

Monin, A., & Obukhov, A. (1954). Basic laws of turbulent mixing in the atmosphere near the ground. vol. 24. *Tr Akad Nauk SSSR Geofiz Inst*, 163–187.

Mooers, G., Pritchard, M., Beucler, T., Ott, J., Yacalis, G., Baldi, P., & Gentine, P. (2021). Assessing the potential of deep learning for emulating cloud superparameterization in climate models with real-geography boundary conditions. *Journal of Advances in Modeling Earth Systems*, *13*(5), e2020MS002385. (e2020MS002385 2020MS002385) doi: https://doi.org/10.1029/2020MS002385

Mooers, G., Tuyls, J., Mandt, S., Pritchard, M., & Beucler, T. G. (2021). Generative modeling of atmospheric convection. In *Proceedings of the 10th international conference on climate informatics* (p. 98–105). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3429309.3429324

Park, S.-B., Gentine, P., Schneider, K., & Farge, M. (2016). Coherent structures in the boundary and cloud layers: Role of updrafts, subsiding shells, and environmental subsidence. *Journal of the Atmospheric Sciences*, *73*(4), 1789–1814.

Perezhogin, P., Zanna, L., & Fernandez-Granda, C. (2023). Generative data-driven approaches for stochastic subgrid parameterizations in an idealized ocean model. *arXiv preprint arXiv:2302.07984*.

Pino, D., de Arellano, J. V.-G., & Duynkerke, P. G. (2003). The contribution of shear to the evolution of a convective boundary layer. *Journal of the atmospheric sciences*, *60*(16), 1913–1926.

Pino, D., Jonker, H. J., Arellano, J. V.-G. d., & Dosio, A. (2006). Role of shear and the inversion strength during sunset turbulence over land: characteristic length scales. *Boundary-layer meteorology*, *121*, 537–556.

Pino, D., & Vilà-Guerau De Arellano, J. (2008). Effects of shear in the convective boundary layer: analysis of the turbulent kinetic energy budget. *Acta Geophysica*, *56*(1), 167.

Priestley, C. H. B., & Swinbank, W. (1947). Vertical transport of heat by turbulence in the atmosphere. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, *189*(1019), 543–561.

Pu, Y., Gan, Z., Henao, R., Yuan, X., Li, C., Stevens, A., & Carin, L. (2016). Variational autoencoder for deep learning of images, labels and captions. *Advances in neural information processing systems*, *29*.

Rasp, S., Pritchard, M. S., & Gentine, P. (2018). Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, *115*(39), 9684–9689.

Salesky, S. T., Chamecki, M., & Bou-Zeid, E. (2017). On the nature of the transition between roll and cellular organization in the convective boundary layer. *Boundary-layer meteorology*, *163*, 41–68.

Shah, S., & Bou-Zeid, E. (2014). Very-large-scale motions in the atmospheric boundary layer educed by snapshot proper orthogonal decomposition. *Boundary-Layer Meteorology*, *153*(3), 355–387.

Shamekh, S., Lamb, K. D., Huang, Y., & Gentine, P. (2022, oct). Implicit learning of convective organization explains precipitation stochasticity. *PNAS*. doi: 10.1002/essoar.10512517.1

Siebesma, A. P., & Cuijpers, J. (1995). Evaluation of parametric assumptions for shallow cumulus convection. *Journal of Atmospheric Sciences*, *52*(6), 650–666.

Siebesma, A. P., Soares, P. M. M., & Teixeira, J. (2007). A combined eddy-diffusivity mass-flux approach for the convective boundary layer. *Journal of the Atmospheric Sciences*, *64*(4), 1230 - 1248. doi: https://doi.org/10.1175/JAS3888.1

Siebesma, A. P., & Teixeira, J. (2000). An advection–diffusion scheme for the convective boundary layer: Description and 1d results. In *Preprints, 14th symp. on boundary layers and turbulence, aspen, co, amer. meteor. soc* (Vol. 133, p. 136).

Smagorinsky, J. (1963). General circulation experiments with the primitive equations: I. the basic experiment. *Monthly weather review*, *91*(3), 99–164.

Soares, P. M. M., Miranda, P. M. A., Siebesma, A. P., & Teixeira, J. (2004). An eddy-diffusivity/mass-flux parametrization for dry and shallow cumulus convection. *Quarterly Journal of the Royal Meteorological Society*, *130*(604), 3365-3383. doi: https://doi.org/10.1256/qj.03.223

Stensrud, D. J. (2009). *Parameterization schemes: keys to understanding numerical weather prediction models*. Cambridge University Press.

Stull, R. B. (1973). Inversion rise model based on penetrative convection. *Journal of Atmospheric Sciences*, *30*(6), 1092 - 1099. doi: https://doi.org/10.1175/1520-0469(1973)030⟨1092:IRMBOP⟩2.0.CO;2

Stull, R. B. (1976). Internal gravity waves generated by penetrative convection. *Journal of Atmospheric Sciences*, *33*(7), 1279–1286.

Stull, R. B. (1988). *An introduction to boundary layer meteorology* (Vol. 13). Springer Science & Business Media.

Sullivan, P. P., & Patton, E. G. (2011). The effect of mesh resolution on convective boundary layer statistics and structures generated by large-eddy simulation. *J. Atmos. Sci.*, *68*, 2395–2415.

Taira, K., Brunton, S. L., Dawson, S. T., Rowley, C. W., Colonius, T., McKeon,

B. J., ... Ukeiley, L. S. (2017). Modal analysis of fluid flows: An overview. *Aiaa Journal*, *55*(12), 4013–4041.

Takida, Y., Liao, W.-H., Lai, C.-H., Uesaka, T., Takahashi, S., & Mitsufuji, Y. (2022). Preventing oversmoothing in vae via generalized variance parameterization. *Neurocomputing*, *509*, 137–156.

Troen, I., & Mahrt, L. (1986). A simple model of the atmospheric boundary layer; sensitivity to surface evaporation. *Boundary-Layer Meteorology*, *37*(1), 129–148.

Wang, W., Huang, Y., Wang, Y., & Wang, L. (2014). Generalized autoencoder: A neural network framework for dimensionality reduction. In *Proceedings of the ieee conference on computer vision and pattern recognition workshops* (pp. 490–497).

Willis, G. E., & Deardorff, J. W. (1974). A laboratory model of the unstable planetary boundary layer. *J. Atmos. Sci.*, *31*, 1297–1307.

Wyngaard, J. C., & Brost, R. A. (1984). Top-down and bottom-up diffusion of a scalar in the convective boundary layer. *Journal of Atmospheric Sciences*, *41*(1), 102–112.

Wyngaard, J. C., & Moeng, C.-H. (1992). Parameterizing turbulent diffusion through the joint probability density. *Boundary-layer meteorology*, *60*(1), 1–13.

Wyngaard, J. C., & Weil, J. C. (1991). Transport asymmetry in skewed turbulence. *Physics of Fluids A: Fluid Dynamics*, *3*(1), 155–162.

Yang, Y., Zheng, K., Wu, C., & Yang, Y. (2019). Improving the classification effectiveness of intrusion detection by using improved conditional variational autoencoder and deep neural network. *Sensors*, *19*(11), 2528.

Yuval, J., & O'Gorman, P. A. (2020). Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions. *Nature communications*, *11*(1), 1–10.

Zhou, B., Sun, S., Sun, J., & Zhu, K. (2019). The universality of the normalized vertical velocity variance in contrast to the horizontal velocity variance in the convective boundary layer. *Journal of the Atmospheric Sciences*, *76*(5), 1437 - 1456. doi: https://doi.org/10.1175/JAS-D-18-0325.1

Zietlow, D., Rolinek, M., & Martius, G. (2021). Demystifying inductive biases for (beta-) vae based architectures. In *International conference on machine learning* (pp. 12945–12954).

linedepth linedepth

Supporting Information for "Learning Atmospheric Boundary Layer Turbulence"

S. Shamekh[1,2], P. Gentine[1,2]

[1]Department of Earth and Environmental Engineering, Columbia University, New York, NY, USA

[2]Center for Learning the Earth with Artificial Intelligence And Physics (LEAP), Columbia University, New York, NY, USA

Contents of this file

1. Figures S1 to S3

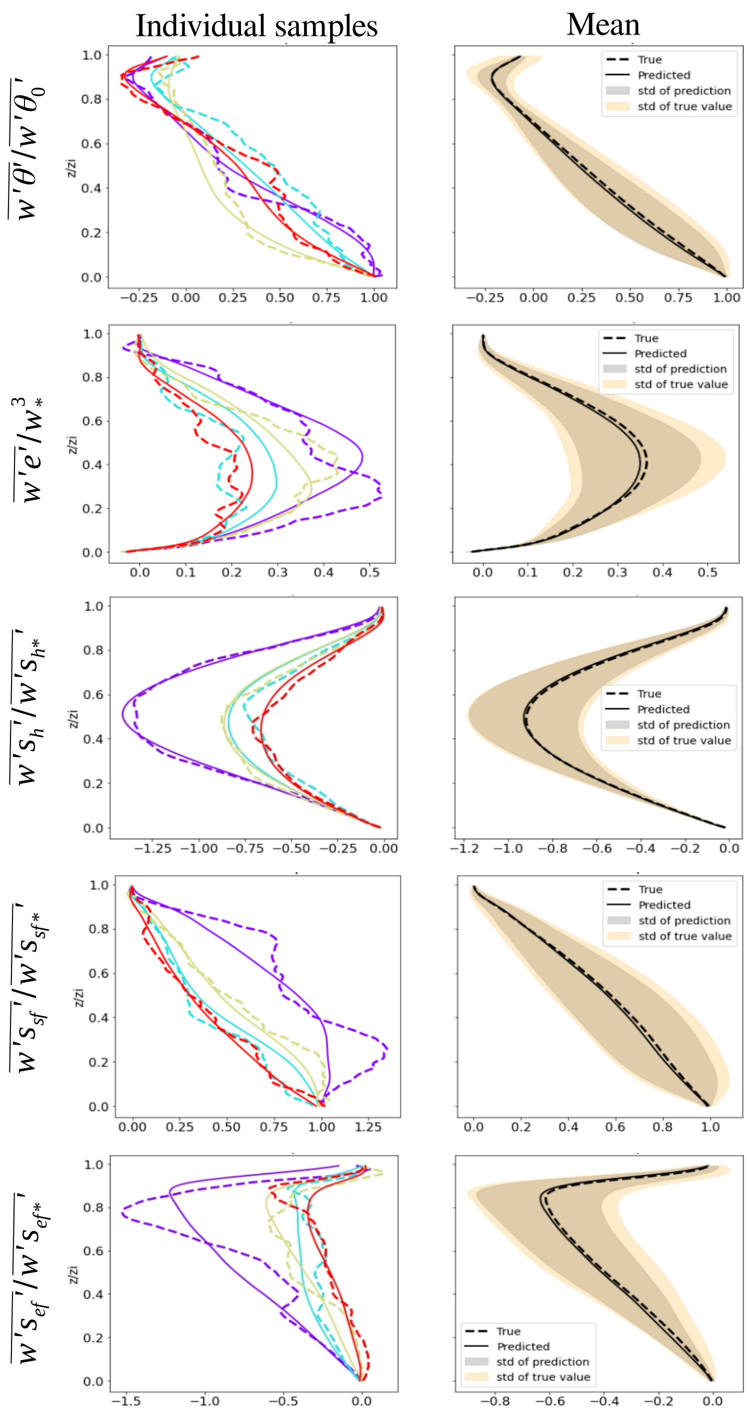


Figure S1: Plots show Flux-NN prediction of scaled vertical turbulent fluxes for simulation 16-0.06. In the left column, the colors distinguish randomly selected samples, displaying the predicted fluxes (line) alongside the true fluxes (dashed line). The right column showcases the mean of the predicted (solid line) and true fluxes (dashed line). Shading indicates the variance in the prediction and true profiles.

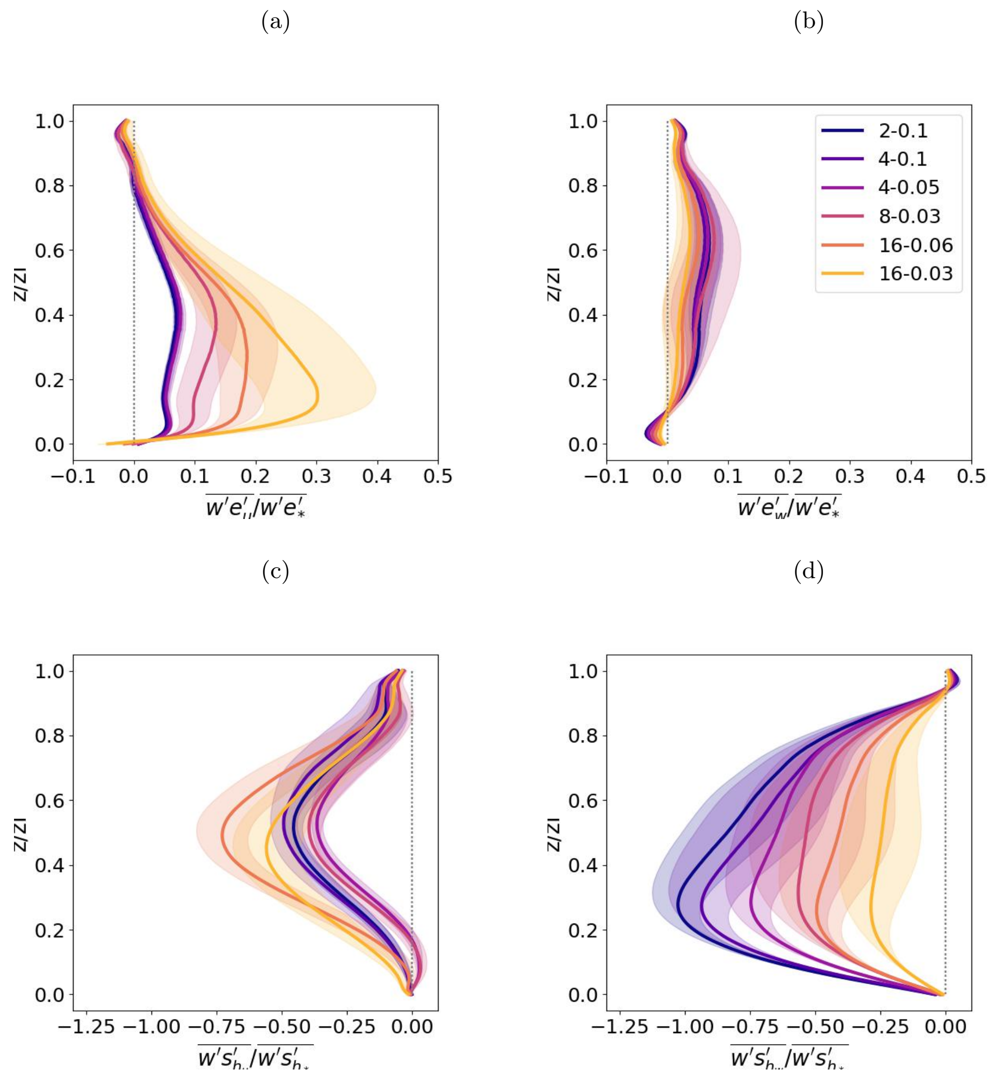


Figure S2: TKE and height dependent tracer flux decomposition to two main modes related to shear and convection.

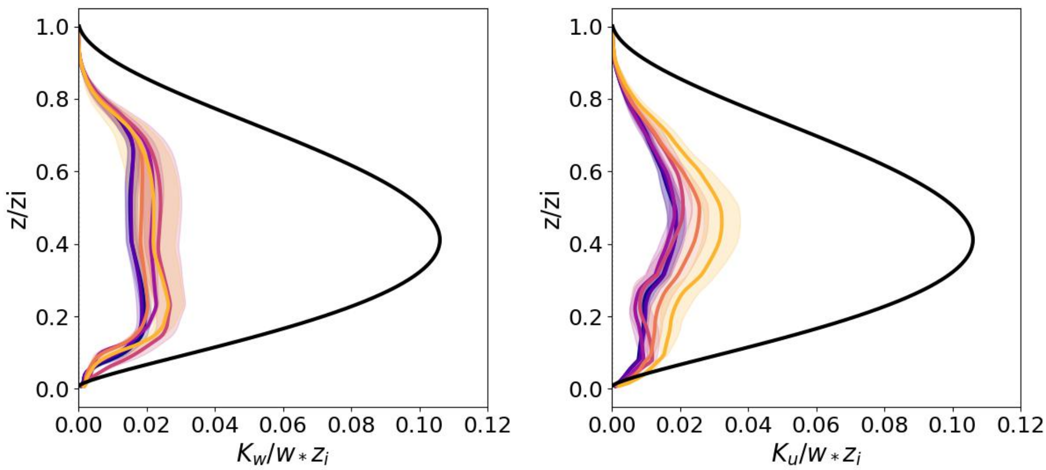


Figure S3: Plot shows (left) convective and (right) shear eddy diffusivity, computed following section 4.4. Black lines shows the eddy diffusivity computed from Holtslag and Moeng (1991) for comparison.