Knowledge-Based Systems 232 (2021) 107458

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Multi-task learning for virtual flow metering

Anders T. Sandnes [a,b,*], Bjarne Grimstad [a,c], Odd Kolbjørnsen [b]

[a] Solution Seeker AS, Oslo, Norway
[b] Department of Mathematics, University of Oslo, Oslo, Norway
[c] Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim, Norway

ARTICLE INFO

Article history:
Received 17 March 2021
Received in revised form 31 August 2021
Accepted 31 August 2021
Available online 2 September 2021

Keywords:
Neural network
Shared parameters
Multi-task learning
Virtual flow metering
Multiphase flow

ABSTRACT

Virtual flow metering (VFM) is a cost-effective and non-intrusive technology for inferring multiphase flow rates in petroleum assets. Inferences about flow rates are fundamental to decision support systems that operators extensively rely on. Data-driven VFM, where mechanistic models are replaced with machine learning models, has recently gained attention due to its promise of lower maintenance costs. While excellent performances in small sample studies have been reported in the literature, there is still considerable doubt about the robustness of data-driven VFM. In this paper, we propose a new multi-task learning (MTL) architecture for data-driven VFM. Our method differs from previous methods in that it enables learning across oil and gas wells. We study the method by modeling 55 wells from four petroleum assets and compare the results with two single-task baseline models. Our findings show that MTL improves robustness over single-task methods, without sacrificing performance. MTL yields a 25%–50% error reduction on average for the assets where single-task architectures are struggling.

1. Introduction

Knowledge of gas, oil, and water flow rates in a petroleum asset is highly valuable in operations and production planning but challenging to obtain [1]. There is a large economic incentive for operators to maintain high production rates and avoid operational problems. Flow rates from individual wells support many important operational decisions, such as production optimization [2], reservoir management [3], and flow assurance [4].

Most assets consist of a set of wells that produce to a shared processing facility, as illustrated in Fig. 1. The joint flow from all wells is continuously measured after being physically separated into its main phases, gas, oil, and water. These can be accurately measured by single phase flow sensors. Flow rates from individual wells are conventionally measured by routing the flow to a dedicated test separator. The resulting observations, known as *well tests*, are of high quality [5]. However, the frequency of well tests is low since the test separator accommodates one well at a time and requires several hours to measure the flow. It is therefore desirable to measure well flow rates before separation.

There are two main strategies for measuring multiphase flow, multiphase flow meters (MPFM) and virtual flow meters (VFM) [6]. MPFMs are complex and expensive measurement devices physically installed in the well. VFM is a soft sensing technology that makes inferences about flow rates from existing sensor data and mathematical models implemented in software. VFM is often seen as complementary to MPFMs. Multiphase flow measurements have higher uncertainty than single phase measurements. Single phase measurements have errors around 0.25% for oil and 1% for gas rates [7]. The quality and availability of water measurements are more varying. A calibrated MPFM is expected to have approximately 5% error for all phases [7]. However, they are specialized to certain operating conditions and must be re-calibrated as conditions change [5].

VFMs can be categorized based on their use of mechanistic or data-driven models [6]. A *mechanistic VFM* is derived from first principles and utilizes empirical correlations sparingly. A *data-driven VFM* is based on a machine learning method that fits a generic mathematical model to data. The generic models do not offer a physical interpretation of the parameters, as opposed to mechanistic models where parameters are related to physical properties. Most VFM solutions today are based on mechanistic models implemented in multiphase flow simulators [8]. There are few, if any, commercially available data-driven VFM solutions. However, there has been an increasing interest in their development [6], which is likely motivated by several factors. First, both instrumentation and data availability have improved. Second, the tooling for machine learning has improved considerably and the number of practitioners has increased. Third, oil and gas profit margins have decreased, leading to a search for more cost-efficient solutions. Data-driven VFM is attractive in terms of cost efficiency due to the promise of low maintenance

* Corresponding author at: Department of Mathematics, University of Oslo, Oslo, Norway.
E-mail address: anders@solutionseeker.no (A.T. Sandnes).

https://doi.org/10.1016/j.knosys.2021.107458

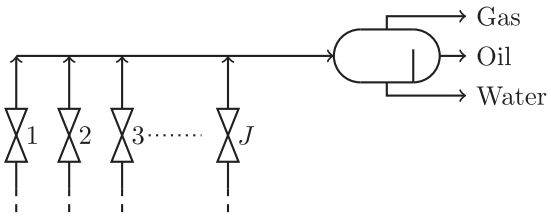

Fig. 1. Asset with J wells sharing a single separator.

requirements and high scalability. Data-driven VFMs are expected to be easier to develop and maintain since they only require flow rate observations to be calibrated [9]. This is in contrast to mechanistic models, which can be challenging to maintain due to high model complexity [10]. Calibration demands flow rate observations and experiment data, such as fluid samples, to attain physically meaningful parameters values. Furthermore, calibration often requires personnel with asset experience and expert knowledge of multiphase flow physics and the VFM software.

A diverse set of methods, models, and experiment setups for data-driven VFM have been presented in the literature. The recent survey in [6] tabulates a selection of the proposed solutions, where architectures based on neural networks are the most frequent. Neural networks have been researched extensively and have been successfully applied in other domains, such as image analysis [11,12], medicine [13], and natural language processing [14], which motivates its popularity in VFM applications. Representative works on neural network based VFMs include [9,15], which report 2.2–4.2% errors and 2.4–4.7% errors respectively. A hybrid solution of neural networks and regression trees is presented in [16], reporting errors in the range of 1.5–6.5%. Gradient boosted trees are explored as an alternative to neural networks in [17], achieving errors of 2%–6% in different scenarios. In all approaches, the VFM model is trained on data from a single well.

Several of the proposed solutions rival the expected performance of conventional MPFMs, but commercially viable alternatives have yet to emerge. The recent study in [18] applied Bayesian neural networks to data-driven VFM. The authors questioned if a robust data-driven method can be obtained by individually modeling wells from historical observations. Several challenges facing any data-driven VFM were highlighted. To reiterate, there are usually few data points for each individual well, making it difficult to identify complex models. Additionally, the underlying process is non-stationary, which makes past data less relevant for future predictions. Finally, the operational practices on most assets may result in low data variety and create highly correlated explanatory variables.

Challenges related to insufficient data are common in machine learning. A solution is to utilize data collected from other related problems [19,20]. There are several ways such data could be combined. One approach is Multi-Task Learning (MTL), where models for all problems are jointly optimized [20,21]. In MTL, the problem is given as a set of tasks, $\{\mathcal{T}_1, \ldots, \mathcal{T}_J\}$, where each task $\mathcal{T}_j$ has a set of observations (y_{ij}, x_{ij}), $i = 1, \ldots, N_j$. MTL attempts to jointly learn models for each task, utilizing the knowledge from other tasks to improve performance. Tasks are assumed to share some common structure that enables the transfer of knowledge. Several mechanisms have been suggested to facilitate knowledge sharing. One approach is to have a set of parameters shared between the task models.

Methods that combine MTL and deep learning have been successfully applied to several domains, e.g., image analysis [22], natural language processing [23], and speech processing [24]. It has also been applied to problems in the energy sector, such as solar and wind power [25,26]. Multi-task neural networks have been presented in a wide range of complexities, from simple feed forward networks [27] to more complex recent architectures that utilize both recurrent and convolutional network components [28]. Some architectures, such as Cross-stitch networks, use a deep neural network for each task and are not designed to scale to numerous tasks [29]. On the opposite side, context-sensitive networks use a task encoding as input to a network with all parameters being shared [30]. A related approach is context adaptation, in which context parameters are learned and used as inputs to a shared neural network [31]. While much work has centered around neural networks, other learners such as support vector machines [32,33] and Gaussian process regression [34] have also been successfully explored.

Even though knowledge sharing has been successful in many cases, it is not guaranteed that all tasks will benefit from each other [35]. Negative transfer refers to the phenomenon where the performance of one task is reduced when another task is introduced. Deciding which task that should be learning together, and how to best avoid negative transfer, is still an open problem.

We present a multi-task learning based data-driven VFM. Our key insight is that knowledge can be shared among wells in a data-driven model, similarly to how knowledge is encoded and reused in mechanistic models. In the context of VFMs, we consider modeling the flow rate from one well as a learning task. Task domains have different data distributions (domain shift) and the tasks must learn different discriminative models. Our MTL architecture, which resembles that of [31], is specialized for data-driven VFM, for which there is a large number of tasks with few observations. It utilizes well-specific parameters to adapt the domains and tasks. Domain adaptation is performed by learning domain-specific feature mappings, which transform input features to abstracted *domain features*. Task adaptation is enabled by learning *task-specific parameters*. The domain features and task parameters are fed to a shared discriminator, to predict flow rates. Because our architecture efficiently scales to many tasks, all wells can be modeled simultaneously.

The framing of data-driven VFM as an MTL problem enables us to learn from more data. While previous methods are limited to small datasets with observations from individual wells, our method scales learning to datasets with observations from any number of wells. To test the proposed method, we perform a study of 55 wells from four assets.

2. Problem description

The system of interest is the well choke valve. Choke valves are adjustable restrictions that are used to control the flow rate from the well. We only consider measurements that are commonly available for oil and gas wells. These are the pressure (p_1) and temperature (T) upstream the choke, the pressure downstream the choke (p_2), and the choke opening (u). In addition, flow rates (q) are measured by a separator (well testing) or an MPFM. A single choke valve is illustrated in Fig. 2.

Flow rates are represented as a vector $q^T = \left[q_G, q_O, q_W\right]$ of gas, oil, and water rate. The rates are customarily given in volumetric flow pr. day, at standard conditions [36]. However, due to the large magnitude of volumetric gas rates, q_G is scaled down by a factor of 1000 to represent liquid equivalents. We denote the total flow rate by $Q = q_G + q_O + q_W$, and the flow composition fractions by $\phi = q/Q$. Flow composition ϕ is dependent on reservoir conditions and is slowly time varying. It can be estimated or assumed fixed between well tests. Here we consider ϕ to be known.

We consider the problem of modeling Q given u, p_1, p_2, T, and ϕ. The gas, oil, and water flow rates are then found as $q = Q\phi$.

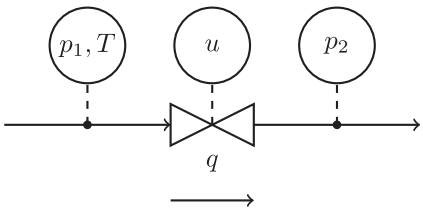

Fig. 2. Choke valve with instrumentation.

2.1. Insights from mechanistic modeling

The system in question poses some challenges that are best explored by a simple mechanistic example. For single phase flow, an analytic model

$$Q = AC\sqrt{\frac{p_1 - p_2}{\rho}}, \tag{1}$$

can be derived from the Bernoulli equation [37]. Here, A is the choke opening area, C is a choke specific flow factor, and ρ is the fluid density. Eq. (1) is the result of generic assumptions and simplifications, and appears in multiple domains. Multiphase extensions to Eq. (1) are usually domain specific. Multiplier models are one class of such extensions for oil and gas flows. They introduce additional factors to Eq. (1) to correct errors in the pressure drop calculation due to multiphase flow. Additionally, the single phase density is replaced by a mixture density. There are several variations of multiplier and density computations, some of which are explored in [38]. These computations often rely on flow composition and fluid properties such as single phase densities.

Eq. (1) contains choke area A as one of the observed variables, and flow factor C as a given constant. However, these quantities are rarely measured directly. In the measurement setup considered here, the choke position is given in percent of full travel. Choke position is not directly comparable between wells, because they have different choke valve designs. It is common to describe choke valves by a CV curve [39]. The CV curve is a mapping between a choke opening and a flow factor, which captures the effect opening area and geometry have on an idealized flow rate. All mechanistic simulators include CV curves or similar mappings. Data-driven approaches often circumvent this by modeling directly on u, which means the mapping is implicit within a black box model.

To utilize a shared discriminator, it is necessary to adapt the observed values to universally comparable quantities and to capture the unique aspects of each well, such as fluid properties and choke geometries.

3. Data

Our data is a set of observations $(Q_{ij}, x_{ij}), i = 1, \dots N_j, j = 1, \dots, J$. Each data point is one observation from one well, indexed as data point i from well j. The total flow rate Q_{ij} is a scalar. Variables x_{ij} is a vector,

$$x_{ij}^\top = \left[u_{ij}, p_{ij,1}, p_{ij,2}, T_{ij}, \phi_{ij,G}, \phi_{ij,O}, \phi_{ij,W}\right], \tag{2}$$

of choke opening, pressure upstream choke, pressure downstream choke, temperature, and flow composition fractions. Observations are taken at time t_{ij}, given in days since the first observation for each well. Time is used for visualization and splitting datasets. We are interested in the steady state behavior of the flow rates. All observation are therefore averages taken over 3–9 h intervals of stable production [40]. Observations are shifted and scaled to lie approximately in the unit interval before model training and evaluation.

3.1. Data exploration

The nature of the underlying process and operational practice can create datasets that are challenging for machine learning models to deal with. For instance, it is common to see reservoir pressure decline as a well develops. As pressure declines, operators will increase the choke opening to keep flow rates stable at a given target. Some assets attempt to reduce the decline, for instance by injecting water into the reservoir [41]. Another remedy is to inject gas into the well flow, which makes the flow composition lighter [42]. Either way, future operating points will generally not be drawn from the same distribution as the training data.

Fig. 3 illustrates the relationship between choke and pressure from all wells, with one well highlighted and colored by time. The systematic development in pressure and operational practice is clear. Models trained on such data are vulnerable to changes in operational practice. A similar pattern can be found in the flow composition, which typically develops into a higher water content with time.

All models trained on data from a single well are vulnerable to correlated explanatory variables and how data change with time. Training on data from multiple wells is one way to overcome these issues. A joint data set has several benefits. The dependencies between explanatory variables become weaker, and the variability within each explanatory variable becomes greater. This is because different wells have different operating regions and operating patterns. The reservoir development also becomes less important. Because, while a single well may move away from its previous operation region, other wells have likely operated under similar conditions before.

The joint data set contains data from 55 wells from four assets. These wells differ in design, operational practice, and reservoir conditions. Fig. 4 explore how the distribution of upstream pressure vary between wells and assets. Many wells have observations in the same range, but one asset is operating at a significantly higher pressure.

4. Model formulation

We propose a data driven virtual flow meter with signature

$$Q_{ij} = f(x_{ij}; \gamma_j, \beta_j, \alpha). \tag{3}$$

It takes input variables x_{ij}, as described in Eq. (2), and is parameterized by three sets of parameters. Two of these parameter sets, γ_j and β_j, are *well specific*, while α is *shared between all wells*. The model is based on a shared neural network. The well specific parameters are used in feature adaptation and task differentiation. The model in Eq. (3) is composed of two steps. A feature adjustment step

$$z_{ij} = g(x_{ij}; \gamma_j), \tag{4}$$

and a flow computation step

$$Q_{ij} = h(z_{ij}; \beta_j, \alpha). \tag{5}$$

The composition is illustrated in Fig. 5.

4.1. Feature adjustment

As discussed in Section 2.1, the observed choke opening is not directly comparable between wells. We are interested in a mapping from u for a universally comparable quantity ψ, which is analog to a CV curve. A piecewise linear mapping is chosen

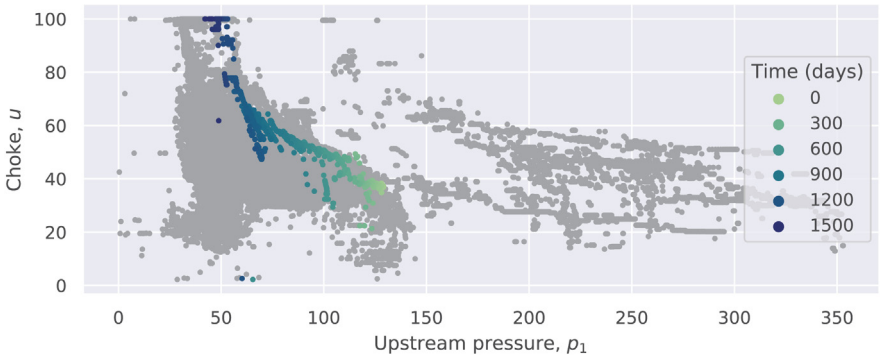

Fig. 3. Scatter plot of choke opening and upstream pressure for all wells. Observations from a single well is highlighted and colored by days since first observation. Choke is continuously adjusted to counteract the declining reservoir pressure.

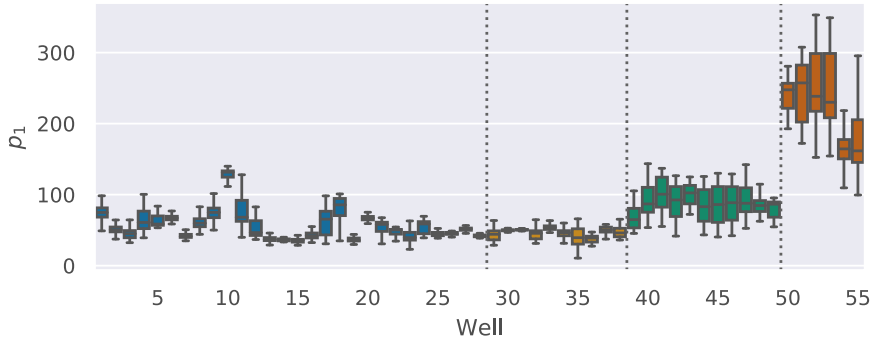

Fig. 4. Box plot of pressure upstream observations for each well. The dotted vertical lines and coloring indicate which wells are from the same asset.

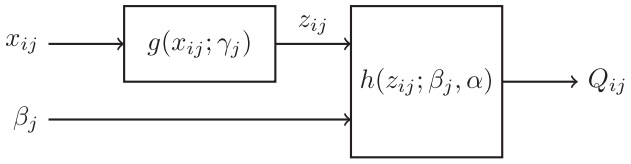

Fig. 5. Block diagram of the model architecture. The model is composed of two functions. A task specific domain adaptation g, and a flow computation h, which takes both task parameters and shared parameters.

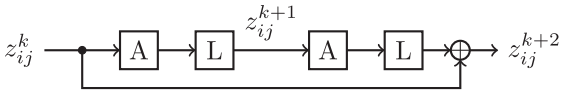

Fig. 6. Diagram of a neural network residual block as described in Eq. (10). The skip connection span two sets of activation (A) and linear layers (L).

for this purpose. It has with m_g break points, $u_1^*, \ldots, u_{m_g}^*$, and is parameterized by $\gamma_j = [\gamma_{j,0}, \ldots, \gamma_{j,m_g}]$. It is formulated as

$$\psi_{ij} = (1+\gamma_{j,0})\left(u_{ij} + \sum_{k=1}^{m_g} \gamma_{j,k} \max(0, u_{ij} - u_k^*)\right), \tag{6}$$

which becomes an identity mapping if all parameters are zero. A monotonic mapping can be enforced by restricting γ_j, but this is not done here. In the examples below we use $m_g = 4$ and set breakpoints to $u_k^* = 0.2k$. Recall that u_{ij} is mapped to the unit interval.

The adjusted feature vector $z_{ij}^\top = [\psi_{ij}, p_{ij,1}, p_{ij,2}, T_{ij}, \phi_{ij,G}, \phi_{ij,O}]$ is then used to evaluate the flow computation. Note that only two of the flow composition fractions are included. This is because the fractions sum to one, and the last component is therefore redundant.

4.2. Flow computation

The flow rate approximation h in Eq. (5) is modeled by a residual feed forward network. The skip connection of the residual blocks spans two hidden layers with pre-activation [43]. There are m_l layers, and all hidden layers have dimension m_h. Linear transforms are parameterized by weights $\alpha = \{(W_k, b_k) | k = 1, \ldots, m_l\}$. The rectifier function $\Phi(z_{ij}^k) = \max(0, z_{ij}^k)$, where the max operation is performed elementwise, is used for activation [21]. There is no activation on the final layer. Adjusted features z_{ij} and task parameters β_j are stacked in a vector before the network is evaluated:

$$z_{ij}^1 = \begin{bmatrix} z_{ij} \\ \beta_j \end{bmatrix}, \tag{7}$$

$$z_{ij}^2 = W_1 z_{ij}^1 + b_1, \tag{8}$$

$$z_{ij}^{k+2} = z_{ij}^k + W_{k+1}\Phi\left(\left[W_k \Phi\left(z_{ij}^k\right) + b_k\right]\right) + b_{k+1}, \quad k = 2, 4, \ldots, m_l - 2, \tag{9}$$

$$Q_{ij} = W_{m_l} z_{ij}^{m_l} + b_{m_l}. \tag{10}$$

The residual blocks in the neural network is illustrated in Fig. 6.

4.3. Model comparison

The effect of multi-task learning is explored by comparing four different models on the given data. Two conventional single-task learning models are used as a baseline. These are compared to two versions of the proposed multi-task architecture.

Both multi-task model formulations are identical, as described above, but they differ in how many tasks are included. The first

option is trained on wells from the same asset. There are four such models because the dataset contains wells from four assets. These are referred to as "MTL-Asset" models. The second option is trained on all wells and is referred to as the "MTL-Universal" model. The two multi-task alternatives are selected to explore the degree of positive and negative transfer between tasks. These models are collectively referred to as the MTL models.

Gradient boosted trees and conventional neural networks are selected as the single-task baselines, as these represent the current state of the art. They are referred to as "STL-GBT" and "STL-ANN" respectively. Gradient boosted trees are based on the description given in [17]. The neural network models are based on the residual architecture described in Section 4.2, but without the task parameters. Both baseline models take all observations except water fraction as input, and total flow as output. An individual copy of each baseline model is identified for each well. These models are only trained and evaluated on data from a single well.

5. Method

Parameters and hyperparameters are found through experimentation and optimization. The dataset is divided into development and test sets. Development data is used to identify hyperparameters and train a final set of models. Test data is only used to evaluate the performance of the final models.

5.1. Data splits

We split the data into subsets used for model development and testing. The development dataset is split further into training and validation sets. Data splits are visualized in Fig. 7.

Test data is selected to reflect how models are used in practice. Meaning that they are trained on all values observed up to a certain point, and then used for weeks or months before they are updated again. For each well, test data is selected such that it comes after development data in time, and the maximum distance between test and development is 120 days. The number of points selected is less than 20% of the observation for that well, and less than 500.

Development data is split further into training and validation. Data points are partitioned into blocks of up to 100 consecutive days. Blocks are randomly divided between training and validation, such that the validation set is 10%–20% of the total.

5.2. Loss and minimization

Model parameters are found by minimizing a standard loss function of prediction error and parameter regularization [44]. All four model types use weighted mean square error with weights w_{ij} as prediction loss.

For the three neural network models (STL-ANN, MTL-Asset, MTL-Universal) the network parameters are regularized by a L_2 norm scaled by a factor λ. We regularize all parameters except the first neural network bias term. For the two MTL models, the task specific parameters are regularized by a L_2 norm scaled by a factor λ_T.

The loss is minimized with the AdamW optimizer [45]. The learning rate is set to 10^{-3}, with a decay rate of 0.5 every 100 of the last 500 epochs. Each optimization runs for 3000 epochs during hyperparameter searches. An additional 1000 epochs are used in the final training. There are three batches pr epoch. Implementation and training are done with PyTorch [46].

STL-GBT models are regularized by penalizing the number of leaves and the squared leaf weight values [17]. Implementation and training are done with XGBoost [47].

5.3. Model evaluation metrics

We use absolute percentage error as the primary performance metric. For data point ij with observed flow rate Q_{ij} and predicted flow rate $\hat{Q}_{ij,M}$ from model M, we find percentage error as $e_{ij,M} = 100(\hat{Q}_{ij,M} - Q_{ij})/Q_{ij}$. For all observations we have $Q_{ij} > 0$. Model subscripts M indicate which of the four model types the error relates to, e.g., MTL-Asset. Root mean squared error is used as a secondary metric.

The mean absolute percentage error (MAPE) for well j with model M is denoted by $E_{j,M}$. Because of the heavy tails of the error distributions, we use a trimmed mean where 5% of the largest errors are removed when computing average errors for individual wells [48].

In addition to the test set performance, we will explore how the models adhere to the expected physical behavior. We expect an isolated increase in upstream pressure to increase flow rate, as indicated by Eq. (1). For any datapoint x_{ij} we have model predictions $\hat{Q}_{ij,M}$. This is compared to $\hat{Q}^{+}_{ij,M}$, which is the same model evaluated on the same data point, with the exception that $p_{ij,1}$ is increased by 10 bar. We compute a binary score

$$s_{ij,M} = \begin{cases} 0 & \text{if } \hat{Q}^{+}_{ij,M} - \hat{Q}_{ij,M} > 0, \\ 1 & \text{otherwise,} \end{cases} \tag{11}$$

to indicate whether this significant increase in pressure also produces an increase in flow rate. The average well score is found as $S_{j,M} = \frac{1}{N_j}\sum_{i=1}^{N_j} s_{ij,M}$. A perfect score, $S_{j,M} = 0$, corresponds to a correct sensitivity to changes in upstream pressure for all data points.

5.4. Hyperparameter selection

Hyperparameters related to model complexity and regularization are optimized for each model individually. E.g., for the STL-ANN models we conduct 55 individual searches. Optimization is done by grid search [49]. In case multiple configurations have similar performance, the one with fewer task or network parameters is preferred.

Neural network models are controlled by the number of hidden layers m_l, hidden layer dimension m_h, and regularization factor λ. Additionally, MTL models require task parameter dimension m_β and task parameter regularization factor λ_T. These parameters are found by grid search, where candidate values are restricted based on the number of data points available for each model type.

STL-GBT models are tuned by the number of leaves, leaf weight, and the number of boosting iterations. These are all found by grid search.

Sample weight w_{ij} is set to 0.1 for multiphase meter observations and 1 for separator observations. This is motivated by the high uncertainty in multiphase meters, as discussed in Section 1. Since most of the data is from multiphase meters, the results are not particularly sensitive to these values. These weights are used for all four model types.

6. Results and discussion

6.1. Test error overview

We first explore how the models generalize by looking at prediction errors across all wells. The results are summarized in Table 1. The performance is quite similar for the three neural network models, but multi-task models are more robust towards large errors. The neural network models outperform STL-GBT. The distribution of prediction errors is heavy tailed, with a few

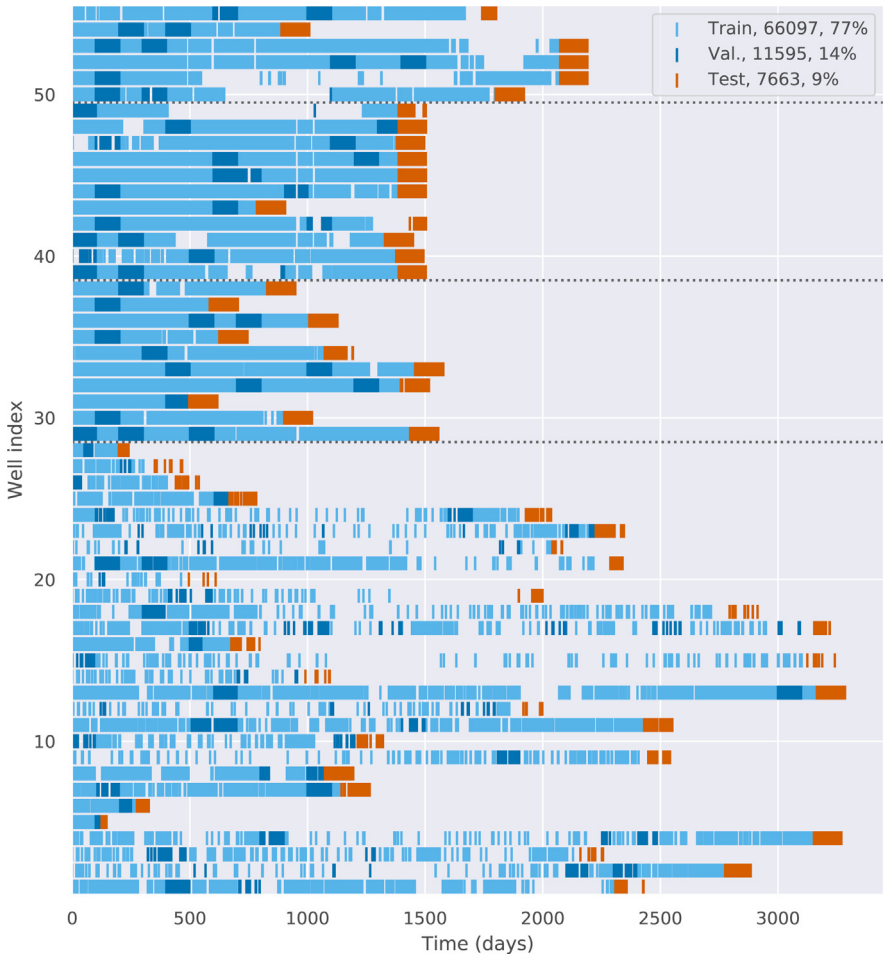

Fig. 7. Training, validation, and test data split for each well. Each mark is one data point. For some wells, the time between observations can be significant. This can be due to long periods with missing measurements, or because the well was closed. Wells from the same asset are grouped by the dotted lines.

Table 1
Summary statistics of absolute percentage error, $|e_{ij,M}|$. Statistics are computed on all test data from all wells. Reported is the mean and a set of percentiles.

Model	Mean	P05	P25	P50	P75	P95
STL-GBT	17.8	0.5	2.9	7.5	16.3	62.3
STL-ANN	20.6	0.4	2.0	4.5	11.2	44.8
MTL-Asset	10.5	0.4	1.8	4.2	9.6	42.1
MTL-Universal	12.8	0.5	2.0	4.4	8.6	33.1

Table 2
Mean prediction errors are computed as trimmed MAPE and RMSE for each well. This yields 55 error estimates for each combination of model and metric, which are summarized by their mean and a set of percentiles.

Metric	Model	Mean	P05	P25	P50	P75	P95
$E_{j,M}^{MAPE}$	STL-GBT	14.5	2.3	5.8	8.6	10.8	53.9
	STL-ANN	10.4	1.4	3.8	5.7	11.1	34.5
	MTL-Asset	8.2	1.4	3.5	6.2	9.0	22.2
	MTL-Universal	7.5	1.6	3.5	5.0	9.2	19.8
$E_{j,M}^{RMSE}$	STL-GBT	9.6	2.0	3.7	6.3	11.0	27.0
	STL-ANN	7.2	1.2	2.5	4.1	8.9	22.4
	MTL-Asset	5.7	1.2	2.5	4.1	7.5	14.1
	MTL-Universal	5.5	0.8	2.5	3.7	6.9	17.1

outliers skewing the mean errors. These outliers motivate the use of trimmed mean when results are reported on a well by well basis.

Fig. 8 illustrates how prediction errors develop with time. As expected, the performance degrades with time for all model types. All model types have great performance in the first few weeks. The benefit of multi-task learning becomes apparent after six weeks.

6.2. Well by well performance

Wells have a different number of data points, and the errors reported in Section 6.1 will naturally be dominated by the wells with many data points. We now explore the test set errors for individual wells. Trimmed MAPE and RMSE values for each well is summarized in Table 2. The three neural network models have similar performance for the best half of the wells, with performance comparable to conventional multiphase meters. Multi-task models are significantly better on the more challenging wells. Neural network models generally outperform STL-GBT. MAPE and RMSE reveal similar patterns. The remainder of the analysis will focus on MAPE values.

6.3. Asset performance

The two MTL models are trained on wells from the same asset and on all wells. To explore the degree of positive and negative knowledge transfer, performance is explored on the four assets. The results are summarized in Table 3. Apart from Asset 3, which has great performance for all neural network models, there is a clear benefit of shared data. For assets 1, 2, and 4, the best multi-task model offers a 25%–50% error reduction compared

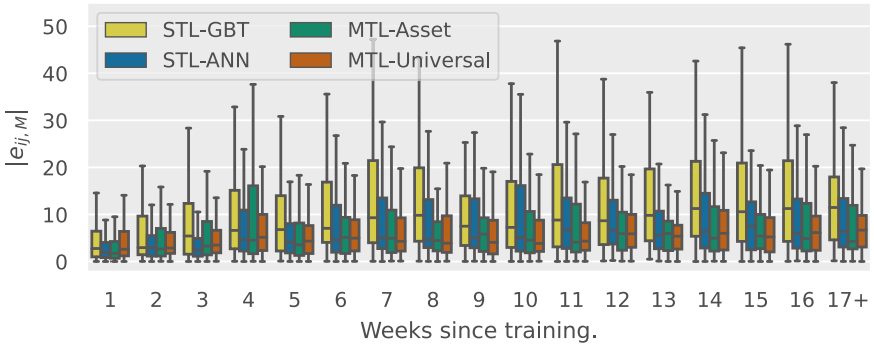

Fig. 8. Box plot of absolute percentage error, grouped by weeks since the last training datapoint. Errors are computed on all test data for all wells.

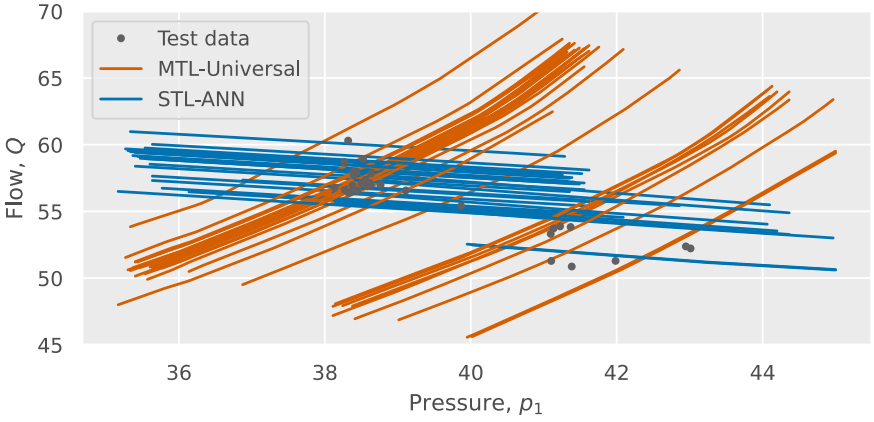

Fig. 9. Comparison of model sensitivity for well 7. Each model is evaluated by taking a subset of 30 test data points (black dots) and varying upstream pressure in a neighborhood around the observed value. The response of STL-ANN is given in blue and MTL-Universal in orange.

Table 3
Model performance grouped by assets. Reported is the average trimmed MAPE for wells from the same asset, for the four model types. The best model type is highlighted.

Model	A. 1	A. 2	A. 3	A. 4
STL-GBT	15.6	13.5	10.4	18.3
STL-ANN	10.9	13.8	5.9	10.5
MTL-Asset	8.1	**10.2**	6.5	7.9
MTL-Universal	**7.3**	11.3	**5.7**	**4.9**

to STL-ANN. However, it is an open question to decide which level of data sharing is best suited for a given well or asset. All model types struggle with Asset 2, which could be due to the limited excitation seen in Fig. 4. Apart from Asset 2, MTL model performance is close to that expected from conventional multiphase meters.

6.4. Sensitivity analysis

Models with great test set performance can still suffer from the data challenges presented in Section 3.1. Correlated explanatory variables make it difficult to isolate the effect of individual variables. Fig. 9 illustrates the issue for Well 7. We expect the response to an increase in upstream pressure to be an increase in flow rate, as indicated by the mechanistic model in Eq. (1). For well 7, both STL-ANN and MTL-Universal models have low test errors, with trimmed MAPE being 2.2% and 1.6% respectively. There is however a significant difference in how they have interpreted the explanatory variables. In this case, the MTL-Universal model was able to identify the expected response, while the STL-ANN was not.

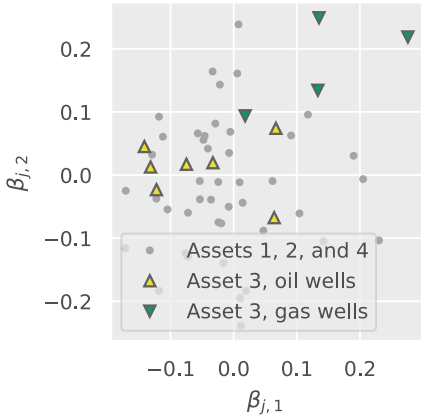

Fig. 10. Plot of β_j from the universal model. Asset 3 has two distinct classes of wells, oil producers and gas producers, which are highlighted by triangle markers.

The observations from Fig. 9 are generalized using the sensitivity metric described in Section 5.3. The results are given in Table 4. A large majority of wells remain unchanged or achieve a better sensitivity score by sharing data with other wells. The advantage of transfer learning is clear in this comparison.

6.5. Ablation study

An ablation study is conducted to better understand the improvements seen in the proposed architecture [50]. The proposed model extends the current state-of-the-art with multi-task learning. Two task adaptation mechanisms are included. These are

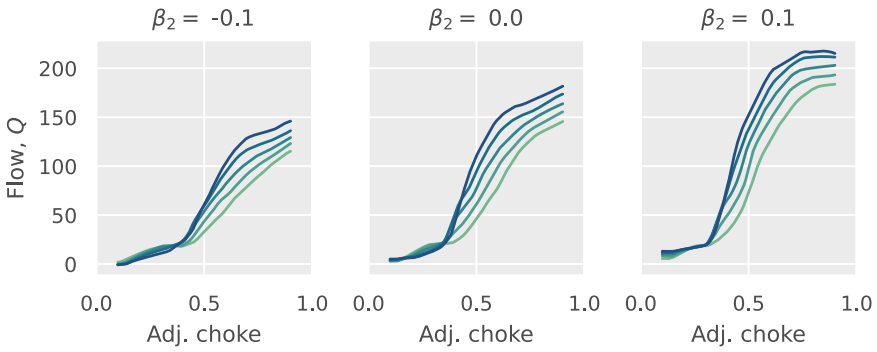

Fig. 11. Effect of β_j on model predictions. The universal model is evaluated on a fixed operating point, except for varying the adjusted choke between 0.1 and 0.9. Each subplot has a fixed value for $\beta_{j,2}$, given in the title. Each curve has a fixed value for $\beta_{j,1}$, ranging from -0.1 for the light green to 0.1 for the dark blue.

Table 4
Mean sensitivity error $S_{j,M}$ for the four model types. Zero is the best score, which means a model had the correct sensitivity for each data point for a given well.

Model	Error
STL-GBT	0.56
STL-ANN	0.28
MTL-Asset	0.14
MTL-Universal	0.07

Table 5
Summary of ablations conducted on the MTL-Universal model. Average trimmed MAPE is computed on all wells. Value for the complete model is repeated from Table 2.

Ablation	Error
Remove γ_j and β_j	10.5
Remove β_j	8.8
Remove γ_j	8.4
Complete model	7.5

task parameters β_j, and domain adaptation parameterized by γ_j. To explore the effect of the task adaptation, the MTL-Universal model is trained with one or both elements removed. When both elements are removed, the model is reduced to a single-task neural network trained on data from all wells. New hyperparameters are found for each ablation. The results is given in Table 5. On average, both adaptation mechanisms have a similar impact. The inclusion of both is beneficial for overall performance, but with diminishing returns, as they are potentially overlapping.

6.6. Model complexity

A multi-task model is more complex than an isolated single-task model. However, as the number of tasks grows, there are several aspects to multi-task learning that leads to overall less complexity. Table 6 summarizes the number of parameters and training time for the four model types presented. STL-GBT is a separate class of models and is much faster to compute than neural networks. For neural network models, the larger models require more time for each model, but less time overall.

In the universal model, the number of well parameters is significantly smaller than the number of parameters needed in an individual well model. On average, an MTL-Asset model requires almost the same number of parameters as MTL-Universal. Indicating that model size does not need to grow significantly with the number of tasks.

In terms of manual work and maintenance, the MTL-Universal formulation scales better with additional wells than conventional model formulations, because it is only one neural network that must be curated. This is highly advantageous for the practical application and commercialization of the results.

Table 6
Summary of model complexity, judged by the number of parameters and time required to train models for all wells. E.g., it took 21 min and 25 s to train the four MTL-Asset models, and they have 711389 parameters in total. All models are trained on a single GPU.

Model	Models	Time	Parameters
STL-GBT	55	00:57	–
STL-ANN	55	56:06	450455
MTL-Asset	4	21:25	711389
MTL-Universal	1	12:38	203851

6.7. Qualitative properties

The selected hyperparameter configuration has two task parameters for each well, $\beta_j^T = [\beta_{j,1}, \beta_{j,2},]$. Fig. 10 illustrates the identified parameters for all wells. Asset 3 has two types off wells, oil producers and gas producers, which have different choke geometries and fluid properties. These wells are separated in space according to their classes, which indicates that task parameters capture physical properties, rather than being proxies for the well index. To further support this, Fig. 11 illustrates how changes in task parameters alter the shape and magnitude of the model response in a consistent fashion.

7. Conclusion

A multi-task learning architecture for data driven virtual flow metering was proposed and explored in a study of 55 wells from four assets. The proposed architecture successfully addresses the identified data challenges, while generally improving model performance. Sharing data between wells improves robustness towards changes in operational practice and makes the model adhere better to the expected physical relationships. In terms of prediction errors, all assets benefit from some level of data sharing. Two of the assets saw average errors reduced by 30%–50% when data was shared between all assets. One asset saw a reduction of 24% when data was shared within the asset, but only 16% improvement when data is shared between all assets. This indicates that issues related to negative transfer could be present in the VFM problem. The final asset saw no significant improvements because the single task architectures already performed well. Overall, the MTL architecture is a promising step towards a data driven virtual flow meter solution.

7.1. Future work

All wells explored here have many data points. It is expected that wells with fewer observations will see a greater benefit from

the knowledge sharing architecture. Exploring these opportunities is left as future work. Additionally, it is desirable to further explore task synergies and negative transfer in the VFM context.

In the proposed model, task specific parameters are constants. In practice, these are likely time varying, since both the well and reservoir will develop over time, e.g., changes in fluid properties, or equipment wear and tear. These aspects are topics for future research.

CRediT authorship contribution statement

Anders T. Sandnes: Conceptualization, Methodology, Software, Investigation, Formal analysis, Visualization, Data curation, Writing – original draft, Writing – review & editing. **Bjarne Grimstad:** Conceptualization, Methodology, Formal analysis, Data curation, Writing – original draft, Writing – review & editing, Supervision. **Odd Kolbjørnsen:** Methodology, Formal analysis, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: The authors Anders T. Sandnes and Bjarne Grimstad are affiliated with the Norwegian company Solution Seeker AS. The company provides a managed AI service to the petroleum industry. Data-driven VFM is part of this service.

Acknowledgment

This work was supported by Solution Seeker Inc. and The Research Council of Norway.

References

[1] L.S. Hansen, S. Pedersen, P. Durdevic, Multi-phase flow metering in offshore oil and gas transportation pipelines: Trends and perspectives, Sensors 19 (9) (2019).
[2] B. Foss, B.R. Knudsen, B. Grimstad, Petroleum production optimization – a static or dynamic problem? Comput. Chem. Eng. 114 (2018) 245–253.
[3] S. Kanshio, A review of hydrocarbon allocation methods in the upstream oil and gas industry, J. Pet. Sci. Eng. 184 (2020).
[4] A.K.M. Jamaluddin, C.S. Kabir, Flow assurance: Managing flow dynamics and production chemistry, J. Pet. Sci. Eng. 100 (2012) 106–116.
[5] S. Corneliussen, J.-P. Couput, E. Dahl, E. Dykesteen, K.-E. Frøysa, E. Malde, H. Moestue, P.O. Moksnes, L. Scheers, H. Tunheim, second ed., Handbook of Multiphase Flow Metering, Norwegian Society for Oil and Gas Measurement, 2005.
[6] T. Bikmukhametov, J. Jäschke, First principles and machine learning virtual flow metering: A literature review, J. Pet. Sci. Eng. 184 (September 2019) (2020) 106487.
[7] R. Thorn, G.A. Johansen, B.T. Hjertaker, Three-phase flow measurement in the petroleum industry, Meas. Sci. Technol. 24 (1) (2012).
[8] A. Amin, Evaluation of commercially available virtual flow meters (VFMs), in: Offshore Technology Conference, Vol. 2, Houston, Texas, USA, 2015, pp. 1293–1318.
[9] T.A. AL-Qutami, R. Ibrahim, I. Ismail, M.A. Ishak, Virtual multiphase flow metering using diverse neural network ensemble and adaptive simulated annealing, Expert Syst. Appl. 93 (2018) 72–85.
[10] B.J. Stenhouse, Modelling and optimisation in BP E&P, in: SPE Intelligent Energy Conference & Exhibition, Vol. 2, Amsterdam, The Netherlands, 2008, pp. 638–645.
[11] J. Yu, M. Tan, H. Zhang, D. Tao, Y. Rui, Hierarchical deep click feature prediction for fine-grained image recognition, IEEE Trans. Pattern Anal. Mach. Intell. (2019).
[12] C. Hong, J. Yu, J. Wan, D. Tao, M. Wang, Multimodal deep autoencoder for human pose recovery, IEEE Trans. Image Process. 24 (12) (2015) 5659–5670.
[13] A.Y. Hannun, P. Rajpurkar, M. Haghpanahi, G.H. Tison, C. Bourn, M.P. Turakhia, A.Y. Ng, Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network, Nature Med. 25 (1) (2019) 65–69.
[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in Neural Information Processing Systems, Vol. 30, Long Beach, CA, 2017.
[15] T.A. AL-Qutami, R. Ibrahim, I. Ismail, M.A. Ishak, Development of soft sensor to estimate multiphase flow rates using neural networks and early stopping, Int. J. Smart Sens. Intell. Syst. 10 (1) (2017) 199–222.
[16] T.A. AL-Qutami, R. Ibrahim, I. Ismail, Hybrid neural network and regression tree ensemble pruned by simulated annealing for virtual flow metering application, in: 2017 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), 2017, pp. 304–309.
[17] T. Bikmukhametov, J. Jäschke, Oil production monitoring using gradient boosting machine learning algorithm, IFAC-PapersOnLine 52 (1) (2019) 514–519.
[18] B. Grimstad, M. Hotvedt, A.T. Sandnes, O. Kolbjørnsen, L.S. Imsland, Bayesian neural networks for virtual flow metering: An empirical study, Appl. Soft Comput. 112 (2021) 107776.
[19] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, G. Zhang, Transfer learning using computational intelligence: A survey, Knowl.-Based Syst. 80 (2015) 14–23.
[20] Y. Zhang, Q. Yang, A survey on multi-task learning, IEEE Trans. Knowl. Data Eng. (2021) 1–20.
[21] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.
[22] C. Hong, J. Yu, J. Zhang, X. Jin, K.-H. Lee, Multimodal face-pose estimation with multitask manifold deep learning, IEEE Trans. Ind. Inf. 15 (7) (2019) 3952–3961.
[23] S. Sigtia, E. Marchi, S. Kajarekar, D. Naik, J. Bridle, Multi-task learning for speaker verification and voice trigger detection, in: ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 2020.
[24] N. Majumder, S. Poria, H. Peng, N. Chhaya, E. Cambria, A. Gelbukh, Sentiment and sarcasm classification with multitask learning, IEEE Intell. Syst. (ISSN: 1941-1294) 34 (3) (2019) 38–43.
[25] Z. Wu, Q. Li, X. Xia, Multi-timescale forecast of solar irradiance based on multi-task learning and echo state network approaches, IEEE Trans. Ind. Inf. 17 (1) (2021) 300–310.
[26] M. Dorado-Moreno, N. Navarin, P.A. Gutiérrez, L. Prieto, A. Sperduti, S. Salcedo-Sanz, C. Hervás-Martínez, Multi-task learning for the prediction of wind power ramp events with deep neural networks, Neural Netw. 123 (2020) 401–411.
[27] R. Caruana, Multitask learning, Mach. Learn. 28 (1) (1997) 41–75.
[28] N. Jin, J. Wu, X. Ma, K. Yan, Y. Mo, Multi-task learning model based on multi-scale CNN and LSTM for sentiment classification, IEEE Access 8 (2020) 77060–77072.
[29] I. Misra, A. Shrivastava, A. Gupta, M. Hebert, Cross-stitch networks for multi-task learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 3994–4003.
[30] D.L. Silver, R. Poirier, D. Currie, Inductive transfer with context-sensitive neural networks, Mach. Learn. 73 (3) (2008) 313–336.
[31] L. Zintgraf, K. Shiarli, V. Kurin, K. Hofmann, S. Whiteson, Fast context adaptation via meta-learning, in: Proceedings of the 36th International Conference on Machine Learning, Vol. 97 of Proceedings of Machine Learning Research, 2019, pp. 7693–7702.
[32] B. Mei, Y. Xu, Safe sample screening for regularized multi-task learning, Knowl.-Based Syst. 204 (2020) 106248.
[33] B. Mei, Y. Xu, Multi-task ν-twin support vector machines, Neural Comput. Appl. 32 (15) (2020) 11329–11342.
[34] Y. Zhou, Y. Liu, D. Wang, G. De, Y. Li, X. Liu, Y. Wang, A novel combined multi-task learning and Gaussian process regression model for the prediction of multi-timescale and multi-component of solar radiation, J. Cleaner Prod. 284 (2021) 124710.
[35] T. Standley, A.R. Zamir, D. Chen, L. Guibas, J. Malik, S. Savarese, Which tasks should be learned together in multi-task learning? in: Proceedings of the 37th International Conference on Machine Learning, Vol. 119 of Proceedings of Machine Learning Research, 2020, pp. 9120–9132.
[36] AIME, Society of Petroleum Engineers, the SI Metric System of Units and SPE Metric Standard, Society of Petroleum Engineers, 1984.
[37] F.M. White, Fluid Mechanics, sixth ed., McGraw-Hill series in mechanical engineering, McGraw Hill, 2008.
[38] R.B. Schüller, T. Solbakken, S. Selmer-Olsen, Evaluation of multiphase flow rate models for chokes under subcritical oil/gas/water flow conditions, SPE Prod. Facil. (ISSN: 1064668X) 18 (3) (2003) 170–181.
[39] A. Grace, P. Frawley, Experimental parametric equation for the prediction of valve coefficient (Cv) for choke valve trims, Int. J. Press. Vessels Pip. 88 (2–3) (2011) 109–118.
[40] B. Grimstad, V. Gunnerud, A. Sandnes, S. Shamlou, I.S. Skrondal, V. Uglane, S. Ursin-Holm, B. Foss, A simple data-driven approach to production estimation and optimization, in: SPE Intelligent Energy International Conference and Exhibition, 2016.
[41] J.J. Sheng, Critical review of low-salinity waterflooding, J. Pet. Sci. Eng. 120 (2014) 216–224.
[42] S. Guet, G. Ooms, Fluid mechanical aspects of the gas-lift technique, Annu. Rev. Fluid Mech. (ISSN: 00664189) 38 (1) (2006) 225–249.

[43] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in: Computer Vision – ECCV 2016, Cham, 2016, pp. 630–645.
[44] T. Hastie, R. Tibshirani, J.H. Friedman, the Elements of Statistical Learning: Data Mining, Inference, and Prediction, second ed., in: Springer series in statistics, Springer, 2009.
[45] D.P. Kingma, J.L. Ba, Adam: A method for stochastic optimization, in: 3rd International Conference on Learning Representations, San Diego, 2015.
[46] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, PyTorch: An imperative style, high-performance deep learning library, Adv. Neural Inf. Process. Syst. 32 (2019).
[47] T. Chen, C. Guestrin, XGBoost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, San Francisco, California, USA, 2016, pp. 785–794.
[48] R.R. Wilcox, Fundamentals of Modern Statistical Methods: Substantially Improving Power and Accuracy, second ed., Springer, 2010.
[49] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, J. Mach. Learn. Res. 13 (2012) 281–305.
[50] Z.C. Lipton, J. Steinhardt, Troubling trends in machine learning scholarship: Some ML papers suffer from flaws that could mislead the public and stymie future research, Queue 17 (1) (2019) 45–77.