



A General Model for Side Information in Neural Networks

Tameem Adel  and Mark Levene * 

Department of Data Science, National Physical Laboratory (NPL), Hampton Road, Teddington TW11 0LW, UK; tameem.adel@npl.co.uk

* Correspondence: mark.levene@npl.co.uk

Abstract: We investigate the utility of side information in the context of machine learning and, in particular, in supervised neural networks. Side information can be viewed as expert knowledge, additional to the input, that may come from a knowledge base. Unlike other approaches, our formalism can be used by a machine learning algorithm not only during training but also during testing. Moreover, the proposed approach is flexible as it caters for different formats of side information, and we do not constrain the side information to be fed into the input layer of the network. A formalism is presented based on the difference between the neural network loss without and with side information, stating that it is useful when adding side information reduces the loss during the test phase. As a proof of concept we provide experimental results for two datasets, the MNIST dataset of handwritten digits and the House Price prediction dataset. For the experiments we used feedforward neural networks containing two hidden layers, as well as a softmax output layer. For both datasets, side information is shown to be useful in that it improves the classification accuracy significantly.

Keywords: side information; neural networks; knowledge base; explainable machine learning

1. Introduction

The use of side information in a supervised machine learning (ML) setting, proposed in [1], was referred to as *privileged information*, and was further developed in [2], who extended the paradigm to unsupervised learning. As illustrated in [3], privileged information refers to data which neither belong to the input space nor to the output space but are still beneficial for learning. In general, side information may, for example, be knowledge given by a domain expert.

In our work, side information comes separate to the training and test data, but unlike other approaches such as the one described above, it can be used by the ML algorithm not only during training but also during testing. In particular, we concentrate on supervised neural networks (NNs) and do not constrain the side information to be fed into the input layer of the network, which constitutes a non-trivial extension of the paradigm. To distinguish between the two approaches we refer to the additional information used in the previous approaches, when the information is only visible during the training phase, as *privileged information*, and in our approach, when the information is visible during both the train and test phases, simply as *side information*. In our model, side information is fed into the NN from a knowledge base, which inspects the current input data and searches for relevant side information in its memory. Two examples of the potential use of side information in ML are the use of azimuth angles as side information for images of 3D chairs [4], and the use of dictionaries as side information for extracting medical concepts from social media [5].

Here, we present an architecture and formalism for side information, and also a proof of concept of our model on a couple of datasets to demonstrate how it may be *useful* in the sense of improving the performance of an NN. We note that side information is not necessarily useful in all cases, and, in fact, it could be *malicious*, causing the performance of an NN to decrease.



Citation: Adel, T.; Levene, M. A General Model for Side Information in Neural Networks. *Algorithms* **2023**, *16*, 526. <https://doi.org/10.3390/a16110526>

Academic Editor: Javier Del Ser Lorente

Received: 24 October 2023

Accepted: 10 November 2023

Published: 15 November 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Our proposal for incorporating side information is relatively simple when side information is available, and, within the proposed formalism, it is straightforward to measure its contribution to the ML. It is also general in terms of the choice of neural network architecture that is used, and able to cater for different formats of side information.

To reiterate, the main research question we address here is when side information comes separate to the training and test data, what is the effect, on the ML algorithm, of making use of such side information, not only during training but also during testing? The rest of the paper is organised as follows: In Section 2, we review some related work, apart from the privileged information already mentioned above. In Section 3, we present a neural network architecture that includes side information, while in Section 4 we present our formalism for side information based on the difference between the loss of the NN without and with side information, where side information is deemed to be useful if the loss decreases. In Section 5, we present the materials and methods used for implementing the proof of concept for side information based on two datasets, while in Section 6 we present the results and discuss them. Finally, in Section 7, we present our concluding remarks.

2. Related Work

We now briefly mention some recent related research. In [6], a language model is augmented with a very large external memory, and uses a nearest-neighbour retrieval model to search for similar text in this memory to enhance the training data. The retrieved information can be used during the training or fine-tuning of neural language models. This work focuses on enhancing language models, and, as such, does not address the general problem of utilising side information.

In a broad sense, our proposal fits into the informed ML paradigm with prior information, where prior knowledge is independent from the input data; has a formal representation; and is integrated into the ML algorithm [7]. Moreover, human knowledge may be input into an ML algorithm in different formats, through examples, which may be quantitative or qualitative [8]. In addition, there are different ways of incorporating the knowledge into an ML algorithm, say an NN, for example, by modifying the input, the loss function, and/or the network architecture [9]. Another trend is that of incorporating physical knowledge into deep learning [10], and more specifically encoding differential equations in NNs, leading to physics-informed NNs [11].

Our method of using probabilistic facts, described in Section 3, is a straightforward mechanism for extracting the information that is needed from the knowledge base in order to integrate them into the neural network architecture in a flexible manner. Moreover, our proposal allows feeding side information into any layer of the NN, not just into the input layer, which is to the best of our knowledge novel.

Somewhat related is also neuro-symbolic artificial intelligence (AI), where neural and symbolic reasoning components are tightly integrated via neuro-symbolic rules; see [12] for a comprehensive survey of the field, and [13] which looks at combining ML with semantic web components. While NNs are able to learn non-linear relationships from complex data, symbolic systems can provide a logical reasoning capability and structured representation of expert knowledge [14]. Thus, in a broad sense, the symbolic component may provide an interface for feeding side information into an NN. In addition, a neuro-symbolic framework was introduced in [15] that supports querying, learning, and reasoning about data and knowledge.

We point out that, as such, our formalism for incorporating side information into an NN does not attempt to couple symbolic knowledge reasoning with an NN as does neuro-symbolic AI [12,15]. The knowledge base, defined in Section 3, acts as a repository, which we find convenient to describe in terms of a collection of facts providing a link, albeit weak, to neuro-symbolic AI. For example, in Section 5, in the context of the well-known MNIST dataset [16], we use expert knowledge on whether an image of a digit is peculiar as side information for the dataset, and inject this information directly into the NN, noting that we do not insist the side information exists for all data items.

3. A General Method for Incorporating Side Information

To simplify the model under discussion, let us assume a generic neural network (NN) with input and output layers and some hidden layers in-between. We use the following terminology: x_t is the input to the NN at time step t , y_t is the true target value (or simply the target value), \hat{y}_t is the estimated target value (or the NN's output value), and s_t is the side information, defined more precisely below.

The side information comes from a *knowledge base*, \mathcal{KB} , containing both *facts* (probabilistic statements) and *rules* (*if-then* statements). To start off, herein, we will restrict ourselves to side information in the form of facts represented as triples, (p, z, c) , with p being the probability that the instance z is a member of class c , and where a class may be viewed as a collection of facts. In other words, the triple (p, z, c) can be written as $p = P(z|c)$, which is the likelihood of z given c . We note that facts may not necessarily be atomic units, and that side information such as s_t is generally viewed as a conjunction of one or more facts.

Let us examine some examples of side information facts, demonstrating that the knowledge base \mathcal{KB} can, in principle, contain very general information:

- (i) $(p, \textit{headache}, \textit{symptom})$ is the probability of a *headache* being a *symptom*. Here, *headache* is a textual instance and the likelihood is $P(\textit{headache}|\textit{symptom})$.
- (ii) $(p, \textit{image}_i, \textit{green})$ is the probability that the background of *image_i* is *green*; in this case the probability could denote the proportion of the background of the image. Here, *image_i* is an image instance and the likelihood is $P(\textit{image}_i|\textit{green})$.
- (iii) $(p, \textit{ecg}_i, \textit{irregular})$ is the probability that *ecg_i* is *irregular*. Here, *ecg_i* is a segment of an ECG time series instance and the likelihood is $P(\textit{ecg}_i|\textit{irregular})$.

The architecture of the neural network incorporating side information is shown in Figure 1. We assume that the NN operates in discrete time steps, and that at the current time step, t , instance x_t is fed into the network and \hat{y}_t is its output. As stated above, we will concentrate on a knowledge base of facts, and detail what preprocessing may occur to incorporate facts into the network as side information. It is important that the preprocessing step at time t can only access the side information and the input presented to the network at that time, i.e., it cannot look into the past or the future. It is, of course, sensible to assume that one cannot access inputs that will be presented to the network in the future at times greater than t , and regarding the past, at times less than t , the reasoning being that this information is already encoded in the network. Moreover, the learner may not be allowed to look into the past due to privacy considerations. On a high level, this is similar to continual learning (CL) frameworks where a CL model is updated at each time step using only the new data and the old model, without access to data from previous tasks due to speed, security, and privacy constraints [17].

As noted in the caption of Figure 1, side information may be connected to the rest of the network in several ways through additional nodes. The decision of whether, for example, to connect the side information to the first or second hidden layers and/or also to the output node can be made on the basis of experimental results. We emphasise that in our model, side information is available in both the training and test phases.

Let x_t be the input instance to the network at time step t and $s_t = \mathcal{KB}(x_t, \mathcal{C})$ represent the side information at time t , where \mathcal{C} is a set of constraints which may be empty. To clarify how side information is incorporated into an NN, we give, in Algorithm 1, high-level pseudo-code corresponding to the architecture for side information, as shown in Figure 1.

Algorithm 1 Training or testing with side information

- 1: x_t is the input to the network at time step t during the train or test phase
- 2: $\lambda \leftarrow$ layer of the network that the side information is fed into
- 3: $\mathcal{C} \leftarrow$ the set of constraints on the knowledge base
- 4: **for** time step $t = 1, 2, \dots, n$ **do**
- 5: x_t is fed as input to the network
- 6: $s_t \leftarrow \mathcal{KB}(x_t, \mathcal{C})$
- 7: an encoding of s_t is fed into the network at layer λ
- 8: $\hat{y}_t \leftarrow$ output of the network
- 9: **end for**

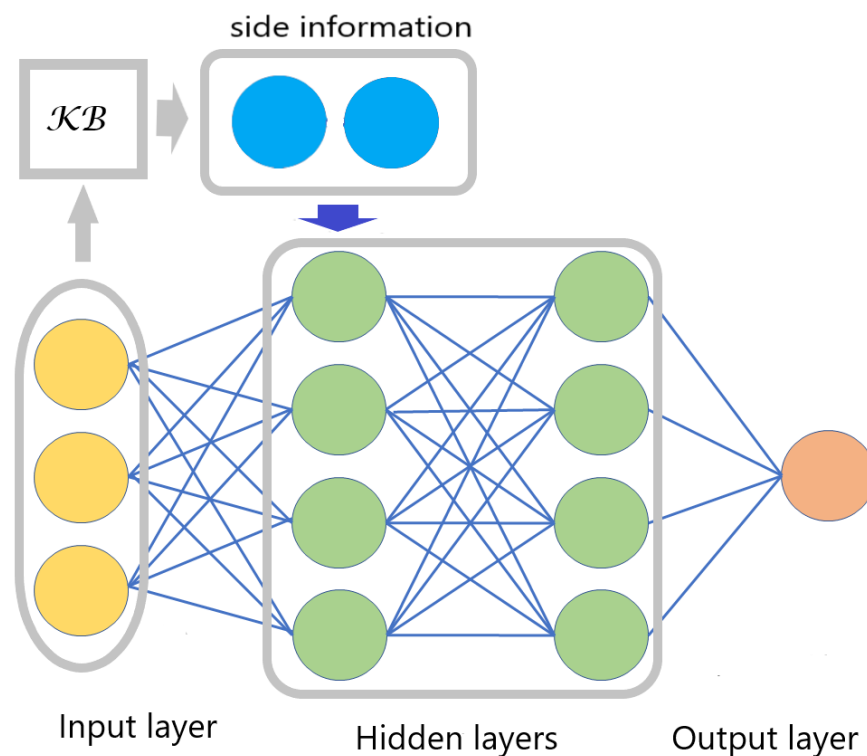


Figure 1. Architecture of a neural network with side information. The arrow from additional side information nodes to the hidden layers indicates that there are weighted links between them. The architecture is instantiated once it is specified which links from side information nodes to the network are active, and furthermore, we do not discount the possibility of having active links directly from side information nodes to the output node.

We stipulate that $\mathcal{KB}(x_t, \mathcal{C}) = \mathcal{KB}_{\mathcal{C}}(x_t)$, i.e., we can either first compute the result of applying \mathcal{KB} to x_t , resulting in $\mathcal{KB}(x_t)$, and then constrain $\mathcal{KB}(x_t)$ to \mathcal{C} , or, alternatively, we can first constrain \mathcal{KB} to \mathcal{C} , resulting in $\mathcal{KB}_{\mathcal{C}}$, and then apply $\mathcal{KB}_{\mathcal{C}}$ to x_t . For this reason, from now on, we prefer to write $\mathcal{KB}_{\mathcal{C}}(x_t)$ instead of $\mathcal{KB}(x_t, \mathcal{C})$, noting that when \mathcal{C} is empty, i.e., $\mathcal{C} = \emptyset$, then $\mathcal{KB}_{\mathcal{C}} = \mathcal{KB}_{\emptyset} = \mathcal{KB}$. We leave the internal working of $\mathcal{KB}_{\mathcal{C}}(x_t)$ unspecified, although we note that it is important that the computational cost of $\mathcal{KB}_{\mathcal{C}}(x_t)$ is low.

The set of constraints \mathcal{C} may be application-dependent, for example, on whether x_t is text- or image-based. Moreover, for a particular dataset we may constrain knowledge-based queries to a small number of one or more classes. To give a more concrete example, if $x_t = image_i$, i.e., the input to the network is an image, and we have constrained the knowledge base to the class *green*, then its output will be of the form $s_t = (p, image_i, green)$. In this case, the knowledge base matches the input image precisely and returns in s_t its

probability of being *green*. In the more general case, the knowledge base will only match part of the input.

More specifically, we let $s_t = \{s_{t1}, s_{t2}, \dots, s_{tm}\}$ be the result of applying the knowledge base \mathcal{KB}_C to the input x_t , where each s_{tj} , for $j = 1, 2, \dots, m$, encodes a fact (p_{tj}, z_{tj}, c_{tj}) , so that the probability, $p_{tj} = P(z_{tj}|c_{tj})$, for $j = 1, 2, \dots, m$, will be fed to the network as side information. From a practical perspective it will be effective to set a non-negative threshold θ such that $s_{tj} = (p_{tj}, z_{tj}, c_{tj})$ is included in s_t only when $p_{tj} > \theta$.

We also assume for convenience that the number of facts returned by $s_t = \mathcal{KB}_C(x_t)$ that are fed into the network as side information is fixed at a maximum, say m , and that they all pass the threshold, i.e., $p_{tj} > \theta$. Now, if there are more than m facts that pass the threshold, then we only consider the top m facts with the highest probabilities, where ties are broken randomly. On the other hand, if there are less than m facts that pass the threshold, say $k < m$, then we consider the values for the $m - k$ facts that did not pass the threshold as missing. A simple solution to deal with the missing values is to consider their imputation [18], and in particular, we suggest using a constant imputation [19] replacing the missing values with a constant such as the mean or the median for the class, which will not adversely affect the output of the network. We observe that if $k = 0$ for all inputs x_t in the test data, but not necessarily in the training data, then side information is reduced to privileged information.

Now, assuming that all the side information is fed directly into the first hidden layer, as is the input data, then it can be shown that using side information is equivalent to extending the input data by m values through a preprocessing step, and then training and testing the network on the extended input. When side information is fed into other layers of the network this equivalence breaks down.

4. A Formalism for Side Information

Our formalism for assessing the benefits of side information is based on quantifying the difference between the loss of the neural network without and with side information; see (1) below. Additionally, we provide detailed analysis for two of the most common loss functions, *squared loss* and *cross-entropy loss* [20]; see (2) and (3), respectively, below.

Moreover, in (5) we present the notion of being *individually useful* at time step t , which is the smallest non-negative δ such that the probability that the difference in loss at t when side information is absent and present is larger than a given non-negative margin, ϵ , which is greater than or equal to $1 - \delta$. When side information is individually useful for all time steps it is said to be *globally useful*. Furthermore, in (6) we modify (5) to allow for the concept of side information being *useful on average*.

To clarify the notation, we assume that x_t is the input to the NN at time t , y_t is the target value given input x_t , \hat{y}_t is the estimated target value given x_t , i.e., the output of the NN when no side information is available for x_t , and \hat{y}_{s_t} is the estimated target value given x_t and side information s_t , i.e., the output of the NN when side information s_t is present for x_t .

To formalise the contribution of the side information, let $\mathcal{L}(y_t, \hat{y}_t)$ be the NN loss on input x_t and $\mathcal{L}(y_t, \hat{y}_{s_t})$ be the NN loss on input x_t with side information s_t . The difference between the loss from \hat{y}_t and the loss from \hat{y}_{s_t} , where $0 \leq y_t \leq 1$ and $0 < \hat{y}_t, \hat{y}_{s_t} < 1$, is given by

$$\mathcal{L}(y_t, \hat{y}_t) - \mathcal{L}(y_t, \hat{y}_{s_t}). \tag{1}$$

Now assuming squared loss for a regression problem, we obtain

$$\begin{aligned} \mathcal{L}(y_t, \hat{y}_t) - \mathcal{L}(y_t, \hat{y}_{s_t}) &= (y_t - \hat{y}_t)^2 - (y_t - \hat{y}_{s_t})^2 \\ &= \hat{y}_t^2 - \hat{y}_{s_t}^2 - 2y_t(\hat{y}_t - \hat{y}_{s_t}) \\ &= (\hat{y}_t + \hat{y}_{s_t})(\hat{y}_t - \hat{y}_{s_t}) - 2y_t(\hat{y}_t - \hat{y}_{s_t}) \\ &= (\hat{y}_t - \hat{y}_{s_t})(\hat{y}_t + \hat{y}_{s_t} - 2y_t), \end{aligned} \tag{2}$$

which has a positive solution when $\hat{y}_t < \hat{y}_{s_t} < 2y_t - \hat{y}_t$, i.e., both terms in the last equation are negative, or $\hat{y}_t > \hat{y}_{s_t} > 2y_t - \hat{y}_t$, i.e., both these terms are positive.

If we restrict ourselves to binary classifiers, then squared loss reduces to the well-known Brier score [21]. Inspecting (2) for this case, when $y_t = 1$ we have the constraint $\hat{y}_t < \hat{y}_{s_t}$, ensuring both terms are negative. Correspondingly, when $y_t = 0$, we have the constraint $\hat{y}_t > \hat{y}_{s_t}$, ensuring that both terms are positive; note that in this case, $\hat{y}_{s_t} < \hat{y}_t$ is intuitive since the classifier returns the probability of y_t being 1, i.e., estimating the chance of a positive outcome.

On the other hand, assuming cross-entropy loss for a classification problem, we obtain

$$\begin{aligned} \mathcal{L}(y_t, \hat{y}_t) - \mathcal{L}(y_t, \hat{y}_{s_t}) &= -y_t \log(\hat{y}_t) + y_t \log(\hat{y}_{s_t}) \\ &= y_t \log\left(\frac{\hat{y}_{s_t}}{\hat{y}_t}\right), \end{aligned} \tag{3}$$

which has a positive solution when $\hat{y}_t < \hat{y}_{s_t}$.

In this case, if we restrict ourselves to binary classifiers, then cross-entropy loss reduces to the well-known logarithmic score [21]. Here, we need to note that a positive solution always exists, since the binary cross-entropy loss is given by

$$-y_t \log(\hat{y}_t) - (1 - y_t) \log(1 - \hat{y}_t), \tag{4}$$

when the estimated target value is \hat{y}_t , and similarly replacing \hat{y}_t with \hat{y}_{s_t} when the estimated target value is \hat{y}_{s_t} .

Now, we would like the probability, $1 - \delta$, for $0 \leq \delta \leq 1$ and a given margin, $\epsilon \geq 0$, for the difference between $\mathcal{L}(y_t, \hat{y}_t)$ and $\mathcal{L}(y_t, \hat{y}_{s_t})$ to be minimised; that is, we wish to evaluate

$$\arg \min_{\delta} P(\mathcal{L}(y_t, \hat{y}_t) - \mathcal{L}(y_t, \hat{y}_{s_t}) \geq \epsilon) \geq 1 - \delta. \tag{5}$$

The probability in (5) can be estimated by parallel or repeated applications of the train and test cycle for the NN without and with side information, and the losses $\mathcal{L}(y_t, \hat{y}_t)$ and $\mathcal{L}(y_t, \hat{y}_{s_t})$ are recorded for every individual run in order to minimise over δ for a given ϵ . Each time the NN is trained its parameters are initialised randomly and the input is presented to the network in a random order, as was done in [22]; other methods of randomly initialising the network are also possible [23].

When (5) is satisfied, then we say that side information is *individually useful* for t , where $t \in \{1, 2, \dots, n\}$. In the case when side information is individually useful for all $t \in \{1, 2, \dots, n\}$ we say that side information is (strictly) *globally useful*. We define side information to be *useful on average* if the following modification of (5) is satisfied:

$$\arg \min_{\delta} P\left(\left(\frac{1}{n} \sum_{t \in \{1, 2, \dots, n\}} \mathcal{L}(y_t, \hat{y}_t) - \mathcal{L}(y_t, \hat{y}_{s_t})\right) \geq \epsilon\right) \geq 1 - \delta, \tag{6}$$

noting that we can estimate (6) in a similar way to estimating (5) with the difference that, in order to minimise over δ for a given ϵ , the average loss is calculated for each run of the NN.

In the case where the greater or equal operator (\geq) in the inequality $\mathcal{L}(y_t, \hat{y}_t) - \mathcal{L}(y_t, \hat{y}_{s_t}) \geq \epsilon$, contained in (5), is substituted with equality to zero ($= 0$) or is within a small margin thereof (i.e., ϵ is close to 0), we may say that side information is *neutral*. Moreover, we note that there may be situations when side information is *malicious*, which can be captured by swapping $\mathcal{L}(y_t, \hat{y}_t)$ and $\mathcal{L}(y_t, \hat{y}_{s_t})$ in (5). We will not deal with these cases herein and leave them for further investigation. (Similar modifications can be made to (6) for side information to be neutral or malicious on average.)

We note that malicious side information may be adversarial, i.e., intentionally targeting the model, or simply noisy, which has a similar effect but is not necessarily deliberate.

In both cases, the presence of side information will cause the loss function to increase. Here, we do not touch upon the methods for countering adversarial attacks, which is an ongoing endeavour [24].

5. Materials and Methods

As a proof of concept of our model of side information, we consider two datasets. The first is the *Modified National Institute of Standards and Technology* (MNIST) dataset [16]. It is a handwritten, 28×28 pixel image, digit database consisting of a training set of 60,000 instances and a test set of 10,000 instances. These are the standard sizes of training and test for MNIST, which result in about 85.7% of the data being used for training and about 14.3% reserved for testing. This is a classification problem where there are 10 labels each depicting a digit identity between 0 and 9.

Our second experiment is based on the House Price dataset, which is adapted from the Zillow's home value prediction Kaggle competition dataset [25], which contains 10 input features as well as a label variable [26]. The dataset consists of 1460 instances. The main task is to predict whether the house price is above or below the median value according to value of the label variable being 1 or 0, respectively, so it is therefore a binary classification problem.

For both experiments we utilised a *fully connected feedforward neural network* (FNN) containing two hidden layers, as well as a softmax output layer; see also Figure 1 and Algorithm 1 regarding the modification of the neural network to handle side information. Our experiments mainly aim at evaluating whether or not the utilisation of side information is beneficial to the performance of the learning algorithm, i.e., whether it is useful. This is achieved by evaluating the difference between the respective NN losses without and with side information. The experiments demonstrate the effectiveness of our approach in improving the learning performance, which is evidenced by a reduction in the loss (i.e., improvements in the accuracy) of the classifiers, for each of the two experiments, during the test phase.

In both experiments, all the reported classification accuracy values and usefulness estimates (according to (5) and (6)) are averaged over 10 separate runs of the NN. For every experiment, we also report the statistical significance values of the differences between the corresponding classification accuracy values, without and with side information. To this effect, we perform a paired *t*-test to assess the difference in accuracy levels obtained, where the null hypothesis states that the corresponding accuracy levels are not significantly different.

For each run out of the 10 repetitions, the parameters of the NN are initialised randomly. In addition, the order via which the input data are presented to the network is also randomly shuffled during each run. Also, missing side information values are replaced with the average value of the side information for the respective class label. The number of epochs used is 10, with a batch size of 250. The adopted loss used in both experiments is the cross-entropy loss.

Before presenting the results in Section 6, we describe some additional detail for the two datasets in Sections 5.1 and 5.2.

5.1. MNIST Dataset

For this dataset we add side information in the form of a single attribute, giving information about how peculiar the handwriting of the corresponding handwritten digit looks. The spectrum of values is between 0, which refers to a very odd-looking digit with respect to its ground-truth identity, and 1, which denotes a very standard-looking digit; we also made use of three intermediate values, 0.25, 0.5 and 0.75. This side information was coded manually for 20% of the test and training data, and was fed into the network through the knowledge base right after the first hidden layer. In accordance with the adopted FNN architecture, we believe that adding side information after the first hidden layer best highlights the utilisation of the proposed concept of side information, since if the side

information was to be added prior to that (i.e., before the first layer), its impact would have been equivalent to extending the input. Otherwise, adding it after the second hidden layer (although technically possible) may not enable the side information to sufficiently impact the learning procedure. According to the (p, z, c) representation, the side information here can be generated as $(p, \text{image}, \text{not-peculiar})$.

As mentioned earlier, an FNN with two hidden layers is utilised. The first hidden layer consists of 350 hidden units, whereas the second hidden layer contains 50 hidden units (these choices are reasonable given the size and structure of the dataset), followed by a softmax activation. Similar to numerous previous works in ML, we focus on comparisons based on an FNN with a certain capacity (two hidden layers in our case, rather than five or six, for instance) in order to concentrate on assessing the algorithm rather than the power of a deep discriminative model.

5.2. House Price Dataset

The FNN utilised in this experiment consists of two hidden layers, followed by a softmax activation. The first hidden layer consists of 12 hidden units, while the second consists of 4 hidden units (these choices are reasonable given the size and structure of the dataset). A portion of 20% of the data is reserved for the test set.

Out of the 10 features, we experimented with one feature at a time being kept aside. Whilst the other 9 features are presented as input to the input layer of the NN in the normal manner, the aforementioned 10th feature is used to generate side information, which is fed to the NN through the knowledge base right after the first hidden layer. The reasoning behind selecting this location for adding the side information is similar to the one highlighted with MNIST. According to the (p, z, c) representation, here, the side information can be generated as $(p, \text{house-data}, \text{feature-value})$, where the feature in question varies according to the one left out.

6. Results and Discussion

For the MNIST dataset, as can be seen in the first row of Table 1, the inclusion of side information according to the manner proposed here leads to a significant improvement in the overall classification accuracy. In addition, the different values of ϵ and δ displayed in the first row in Table 2 indicate that, with MNIST, side information is useful, both individually and on average. Due to the fact that MNIST is a dataset on which FNNs typically attain high accuracy values, achieving an improvement when learning from MNIST is important, even if it is not so large.

For the House Price dataset, as can be seen in the second and third rows of Table 1, including a single feature as side information always leads to a significant improvement in the classification accuracy. We refer to Group 1 of the features as the scenario of leaving out one of the first five features (i.e., feature 1, 2, 3, 4 or 5) as the side information feature. Correspondingly, we refer to Group 2 as the scenario based on leaving out one of the other five features (i.e., feature 6, 7, 8, 9 or 10) as the side information feature. We have found it convenient to group features in this manner given the fact that those belonging to the same group have ultimately resulted in very similar accuracy values. Interestingly, we notice that using any of the first five features as the side information feature (in Group 1) leads to a higher improvement than leaving out one of the latter five features (in Group 2). As such, information involved in the first five features is seemingly more relevant for the classifier. This is also in line with the intuition that can be obtained after a manual inspection of the features, since the information involved in the Group 1 features is overall more global than the information in the Group 2 features. For example, Group 1 contains information about the overall quality and condition of the house. This opens possibilities for utilising our side information framework for future work on explainability [27] as well as feature selection [28].

The values of ϵ and δ reported in the second and third rows of Table 2 demonstrate the fact that including side information in the learning procedure is considerably useful, both

individually and on average, when learning from the House Price dataset. Furthermore, the findings concluded from Table 2 are in line with those from Table 1 regarding the dominance of the features from Group 1 over the Group 2 features in terms of relevance.

Table 1. Test classification accuracy for the two experiments: MNIST and House Price, followed by the results of a statistical significance test (p -value for a 95% confidence interval). A bold entry indicates that the classification accuracy of the FNN for the respective experiment is significantly higher than its competitor. The inclusion of side information leads to a statistically significant improvement in the classification accuracy in both experiments, compared to the case when side information is not included.

Classification Accuracy	Without Side Information	With Side Information	p -Value
MNIST	97.85%	98.01%	0.044
House Price (Group 1)	50.10%	54.28%	0.000173
House Price (Group 2)	50.40%	53.43%	0.0012

Table 2. Difference between the classification losses for the two experiments: MNIST and House Price, without and with side information, expressed in the second and third columns by ϵ and δ , according to (5) when side information is individually useful (we note that the ϵ values were normalised and reported as percentages). In addition, in the fourth and fifth columns, the values of ϵ and δ express how useful side information is on average, according to (6). For both datasets, side information is useful both individually and on average.

Usefulness of Side Information	Individually Useful		Useful on Average	
	ϵ	δ	ϵ	δ
MNIST	0.06%	0.20	0.09%	0.20
House Price (Group 1)	4.10%	0.10	4.80%	0.10
House Price (Group 2)	2.96%	0.10	3.60%	0.10

7. Concluding Remarks

Herein, we presented a novel model for side information in neural networks, where the side information is made visible to the network during both the training and testing phases. The side information is fed into the neural network with the aid of a knowledge base of facts, as shown in Figure 1 and the high-level pseudo-code given in Algorithm 1.

We developed a formalism for side information, whereby it can be individually useful according to (5) or useful on average according to (6). As a proof of concept, we then provided experimentation on two datasets: MNIST and House Price, averaging the results over 10 runs of an FNN with two hidden layers. For both datasets the use of side information improves the classification accuracy significantly and is useful both individually and on average. Extensive experimentation to provide further validation of the approach is a task that is left as further work.

In future work, we aim to investigate the relationship between side information and explainable ML models, noting that explainability is one of the pillars of trustworthy ML, broadly known as explainable AI (XAI) [27]. By explainable ML, we refer to methods which aim to analyse the data in a way that involves attributes that can be understandable to humans. The peculiarity of digits in the MNIST dataset is an example of such an attribute. Some of these attributes can be cast as side information in our formalism, and when they are shown to be useful will improve the performance of the NN.

Author Contributions: Both authors contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the UK Government’s Department for Science, Innovation and Technology (DSIT).

Data Availability Statement: The datasets presented in this study are openly available. The MNIST dataset is available at <http://yann.lecun.com/exdb/mnist/> (accessed on 9 November 2023), and the House Price dataset is available at <https://www.kaggle.com/datasets/moewie94/housepricedata> (accessed on 9 November 2023).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Vapnik, V.; Vashist, A. A new learning paradigm: Learning using privileged information. *Neural Netw.* **2009**, *22*, 554–557. [[CrossRef](#)] [[PubMed](#)]
2. Shekhar, S.; Akoglu, L. Incorporating privileged information to unsupervised anomaly detection. In Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), Dublin, Ireland, 10–14 September 2019; pp. 87–104.
3. Jonschkowski, R.; Hofer, S.; Brock, O. Patterns for learning with side information. *arXiv* **2015**, arXiv:1511.06429.
4. Adel, T.; Ghahramani, Z.; Weller, A. Discovering interpretable representations for both deep generative and discriminative models. In Proceedings of the 35th International Conference on Machine Learning (ICML), Stockholm, Sweden, 10–15 July 2018; pp. 50–59.
5. Hasan, A.; Levene, M.; Weston, D. Learning structured medical information from social media. *J. Biomed. Inform.* **2020**, *110*, 103568. [[CrossRef](#)] [[PubMed](#)]
6. Borgeaud, S.; Mensch, A.; Hoffmann, J.; Cai, T.; Rutherford, E.; Millican, K.; van den Driessche, G.; Lespiau, J.-B.; Damoc, B.; Clark, A.; et al. Improving language models by retrieving from trillions of tokens. In Proceedings of the 39th International Conference on Machine Learning (ICML), Baltimore, MA, USA, 17–23 July 2022; Volume 162, pp. 2206–2240.
7. von Rueden, L.; Mayer, S.; Beckh, K.; Georgiev, B.; Giesselbach, S.; Heese, R.; Kirsch, B.; Pfrommer, J.; Pick, A.; Ramamurthy, R.; et al. Informed machine learning—A taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 614–633. [[CrossRef](#)]
8. Deng, C.; Ji, X.; Rainey, C.; Zhang, J.; Lu, W. Integrating machine learning with human knowledge. *iScience* **2020**, *23*, 27. [[CrossRef](#)] [[PubMed](#)]
9. Dash, T.; Chitlangia, S.; Ahuja, A.; Srinivasan, A. A review of some techniques for inclusion of domain-knowledge into deep neural networks. *Nat. Sci. Rep.* **2022**, *12*, 1040. [[CrossRef](#)] [[PubMed](#)]
10. Monaco, S.; Apiletti, D.; Malnati, G. Theory-guided deep learning algorithms: An experimental evaluation. *Electronics* **2022**, *11*, 2850. [[CrossRef](#)]
11. Cuomo, S.; Di Cola, V.; Giampaolo, F.; Rozza, G.; Raissi, M.; Piccialli, F. Scientific machine learning through physics-informed neural networks: Where we are and what is next. *J. Sci. Comput.* **2022**, *92*, 62. [[CrossRef](#)]
12. Wang, W.; Yang, Y.; Wu, F. Towards data-and knowledge-driven artificial intelligence: A survey on neuro-symbolic computing. *arXiv* **2022**, arXiv:2210.15889.
13. Breit, A.; Waltersdorfer, L.; Ekaputra, F.; Sabou, M.; Ekelhart, A.; Iana, A.; Paulheim, H.; Portisch, J.; Revenko, A.; Teije, A.T.; et al. Combining machine learning and semantic web: A systematic mapping study. *ACM Comput. Surv.* **2023**, *55*, 41. [[CrossRef](#)]
14. Prentzas, J.; Hatzilygeroudis, I. Exploring aspects regarding reasoning in neuro-symbolic rules and connectionist expert systems. In Proceedings of the 12th International Conference on Information, Intelligence, Systems & Applications (IISA), Chania Crete, Greece, 12–14 July 2021; p. 8.
15. Badreddine, S.; Garcez, A.; Serafini, L.; Spranger, M. Logic tensor networks. *Artif. Intell.* **2022**, *303*, 39. [[CrossRef](#)]
16. LeCun, Y.; Bottou, L.; Bengio, Y. Gradient-based learning applied to document recognition. *Proceeding IEEE* **1998**, *86*, 2278–2324. [[CrossRef](#)]
17. Adel, T.; Zhao, H.; Turner, R. Continual learning with adaptive weights (CLAW). In Proceedings of the 8th International Conference on Learning Representations (ICLR), Addis Ababa, Ethiopia, 26–30 April 2020; p. 19.
18. van Buuren, S. *Flexible Imputation of Missing Data*; Interdisciplinary Statistics Series; Chapman & Hall/CRC: Boca Raton, FL, USA, 2012.
19. Perez-Lebel, A.; Varoquaux, G.; Le Morvan, M.; Josse, J.; Poline, J. Benchmarking missing-values approaches for predictive models on health databases. *GigaScience* **2022**, *11*, 22. [[CrossRef](#)] [[PubMed](#)]
20. Murphy, K. *Probabilistic Machine Learning: An Introduction*; Adaptive Computation and Machine Learning; MIT Press: Cambridge, MA, USA, 2022.
21. Bröcker, J.; Smith, L. Scoring probabilistic forecasts: The importance of being proper. *Weather Forecast.* **2007**, *22*, 382–388. [[CrossRef](#)]

22. Lakshminarayanan, B.; Pritzel, A.; Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In Proceedings of the Advances in Neural Information Processing Systems (NIPS) 30, Long Beach, CA, USA, 4–9 December 2017; p. 12.
23. Narkhede, M.; Bartakke, P.; Sutaone, M. A review on weight initialization strategies for neural networks. *Artif. Intell. Rev.* **2022**, *55*, 291–322. [[CrossRef](#)]
24. Bai, T.; Luo, J.; Zhao, J.; Wen, B.; Wang, Q. Recent advances in adversarial training for adversarial robustness. In Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21), Montreal, QC, Canada, 19–27 August 2019; pp. 4312–4321.
25. Zillow Prize. Zillow’s Home Value Prediction (Zestimate). 2017. Available online: <https://www.kaggle.com/c/zillow-prize-1/data> (accessed on 9 November 2023).
26. Ngo, T. House Price Data. 2020. Available online: <https://www.kaggle.com/datasets/moewie94/housepricedata> (accessed on 14 November 2023).
27. Rawal, A.; McCoy, J.; Rawat, D.; Sadler, B.; St. Amant, R. Recent advances in trustworthy explainable artificial intelligence: Status, challenges and perspectives. *IEEE Trans. Artif. Intell.* **2022**, *3*, 852–866. [[CrossRef](#)]
28. Venkatesh, B.; Anuradha, J. A review of feature selection and its methods. *Cybern. Inf. Technol.* **2019**, *19*, 3–26. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.