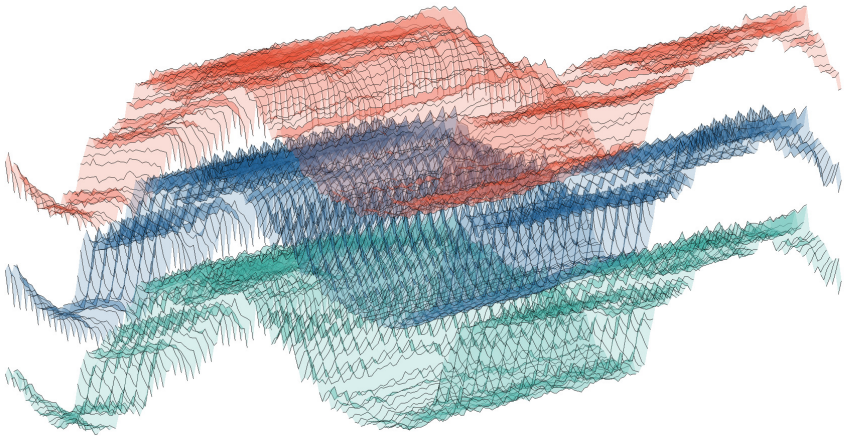


En jämförelse av AI-modeller för kortsiktig lastprognostisering

Utveckling och utvärdering av AI-modeller vid prognostisering av elektrisk last i ett småskaligt energisystem

Kandidatarbete vid Institutionen för Elektroteknik

Ludvig Eriksson, Jakob Johansson, Richard Johnsson, Lasse Kötz, Johan Lamm, Ellinor Lundblad

KANDIDATARBETE 2021:VT

En jämförelse av AI-modeller för kortsiktig lastprognostisering

Utveckling och utvärdering av AI-modeller vid prognostisering av elektrisk last i ett småskaligt energisystem

Ludvig Eriksson, Jakob Johansson, Richard Johnsson, Lasse Kötz, Johan Lamm, Ellinor Lundblad

CHALMERS
UNIVERSITY OF TECHNOLOGY

Institutionen för Elektroteknik
Avdelningen Elkraftsteknik
EENX15
Chalmers Tekniska Högskola
Göteborg 2021

En jämförelse av AI-modeller för kortsiktig lastprognostisering
Utveckling och utvärdering av AI-modeller vid prognostisering av elektrisk last i ett småskaligt energisystem

Ludvig Eriksson
Jakob Johansson
Richard Johnsson
Lasse Kötz
Johan Lamm
Ellinor Lundblad

Handledare:
Kyriaki Antoniadou-Plytaria, Doktorand, Instutitionen för elektroteknik
Nima Mirzaei Alavijeh, Doktorand, Instutitionen för elektroteknik
David Steen, Doktor, Instutitionen för elektroteknik

Kandidatarbete 2021: VT
Institutionen för Elektroteknik
Avdelningen Elkraftsteknik
EENX15
Chalmers Tekniska Högskola
Göteborg 2021

Omslag: Resultatet ifrån prognostiseringen med de tre modellerna presenterat i ett 3D-formaterat färgdiagram.

Förord

Först och främst vill vi tacka våra handledare, Nima Mirzaei Alavijeh, Kyriaki Antoniadou-Plytaria doktorander i elektroteknik och David Steen doktor i elektroteknik. Genom de senaste månaderna har vi alltid kunnat räkna med ert stöd oavsett om vi har träffat på hinder eller behövt respons på våra angreppsmetoder.

Vidare vill vi även rikta ett tack till doktor Azam Bagheri för givande respons på vårt praktiska tillvägagångssätt för träning och uppbyggnad av maskininlärningsmodellerna.

Vi vill också tacka HSB Living Lab för tillgången till den elektriska lastdata som varit nödvändig för att utföra arbetet.

Comparison of AI models for short-term load forecasting
Development and evaluation of AI models for electric load forecasting on a small scale energy system

Ludvig Eriksson, Jakob Johansson, Richard Johnsson, Lasse Kötz,
Johan Lamm, Ellinor Lundblad

Department of Electrical Engineering
Chalmers University of Technology

Abstract

Based on the research community's demand for short-term electric load forecasting, this project aims to build and evaluate the performance of three different machine learning models in their prediction abilities. The models studied are linear regression (LR), artificial neural network (ANN) and recurrent neural network (RNN) with long short-term memory (LSTM). The models were trained for a forecasting horizon of 24 hours with a time discretization step of 15 minutes. Historical electric load measurements, obtained from the HSB Living lab (a building of 29 residential apartments), were used as the training dataset, while open-source Python programming libraries for machine learning were used to develop and train the models. The correlation of the electric load with other data types, such as weather data from SMHI, was also investigated. However, the investigation showed little to no relevance of any data other than the historical load. After training and optimization, the model evaluation indicated a similar ability to provide short-term electric load forecasting performance, with about 12% Mean Absolute Percentage Error (MAPE) for all three models. Moreover, it was concluded that the ANN and RNN LSTM models were not able to fully identify non-linear patterns needed to outperform the LR model for this small-scale energy system.

Keywords: Short-term load forecasting; LSTM; ANN; Linear regression; machine learning; AI; small scale; apartment building;

En jämförelse av AI-modeller för kortsiktig lastprognostisering
Utveckling och utvärdering av AI-modeller vid prognostisering av elektrisk last i ett småskaligt energisystem

Ludvig Eriksson, Jakob Johansson, Richard Johnsson, Lasse Kötz,
Johan Lamm, Ellinor Lundblad

Institutionen för Elektroteknik
Chalmers Tekniska Högskola

Sammandrag

Baserat på forskarsamfundets behov av kortsiktiga elektriska lastprognostiseringar, syftar detta projekt till att bygga och evaluera prestationen av tre olika maskininlärningsmodeller i deras prognostiseringsförmågor. Modellerna är linjär regression (LR), artificiellt neuronnät (ANN) och återkommande neuronnät (RNN) med långt korttidsminne (LSTM). De anpassades för en prognostiseringshorisont på 24 timmar med 15 minuters tidsupplösning. Datan som användes för modellerna var historisk elektrisk last ifrån HSB living lab (flerbostadshus med 29 lägenheter). Byggandet och träningen av modellerna gjordes i programmeringsspråket Python med programbibliotek kopplade till maskininlärning som har öppen källkod. Korrelation för elektrisk last mot andra typer av indata som väderdata ifrån SMHI undersöktes. Denna undersökning visade dock en ytterst liten till obefintlig relevans för annan data än historisk last. Efter träning och optimering av modellerna hade de liknande kvalitéer i sin lastprognostiseringsförmåga, cirka 12% MAPE (eng. Mean Absolute Percentage Error) vid korsvalidering. Slutsatsen som kunde dras var att ANN- och RNN LSTM-modellerna inte kunde identifiera icke-linjära mönster tillräckligt bra för att överträffa LR-modellen i detta småskaliga energisystem.

Nyckelord: Kortsiktig elektrisk lastprognostisering, LSTM, ANN, linjär regression, AI, småskalig, lägenhetshus

Ordlista

ANN	Förkortning för artificiellt neuronnät (eng. Artificial Neural Network) och syftar i denna rapport till ett bakåtpropagerande neuronnät.
Epok	En träningskörning av modellen. Avslutas då modellen har sett all träningsdata. Fler epoker ger modellen möjlighet att anpassas bättre till träningsdatan.
LR	Förkortning för linjär regression.
LSTM	Förkortning av långt korttidsminne (eng. Long Short-Term Memory).
MAPE	Förkortning för medelvärdet av det absoluta procentuella felet (eng. Mean Absolute Percentage Error).
MSE	Förkortning för medelvärdet av det kvadratiska felet (eng. Mean Square Error).
RNN	Förkortning för återkommande neuronnät (eng. Recurrent Neural Network).
Seriestorlek	(eng. Batch Size) Anger hur mycket data som modellen ges åt gången. En serie är en del av en epok, men epoken är inte färdig förrän modellen kört så många serier att all träningsdata har behandlats.
Tidshorisont	Anger hur långt fram i tiden prognosen gäller för.
Tidsupplösning	Anger storleken på tidsintervallen mellan prognosens värden. En prognos där värden förutspås för varje timme framåt i 24 timmar har en tidsupplösning på en timme och en tidshorisont på 24 timmar.

Innehåll

1 Inledning

Den globala uppvärmningen och dess påverkan på klimatet är en av vår tids största utmaningar [1]. Det går tydligt att se storleken på problemet och dess relevans genom världens länders engagemang för att axla utmaningen. Ett konkret exempel på detta är undertecknandet av Parisavtalet 2016, som innehåller ett flertal klimatmål och visar på en eftersträvan att uppnå dessa [2].

En av de största nycklarna till att nå de klimatmål som ingår i Parisavtalet är att säkra elproduktionen med förnyelsebara energikällor [3], exempelvis sol- och vindenergi. Det finns dock en utmaning med att använda exempelvis sol- och vindenergi i energiproduktion på grund av dess variationer i produktion. Sol- och vindenergi är intermittenta energikällor där produktion inte kan styras, utan främst beror på den aktuella solinstrålningen respektive aktuella vindförhållanden [4], [5].

Det finns olika strategier för att komma runt problemet med att elen inte alltid produceras när den behövs. Energi kan lagras eller så kan icke-tidsberoende elanvändning schemaläggas när det finns ett överflöd av energi. Småskaliga elproducenter som i huvudsak producerar för eget bruk, exempelvis privatpersoner med solceller, kan även välja att till exempel sälja el tillbaka till elnätet, med fördel då behovet är stort och elpriset högt [4]. För att optimera denna typ av beslut och planering är det till stor hjälp att ha träffsäkra prognoser för hur elförbrukning, elproduktion och elpriser kommer se ut.

Ett välanvänt verktyg inom många olika områden, inklusive prognostisering, är artificiell intelligens (AI). AI är ett hett ämne idag [6] och används till många olika ändamål och syften, t.ex. inom datorspels-, fordons-, finansindustrin, och annonsverksamhet [7], [8].

Konceptet AI har funnits länge, men har på grund av sin komplexitet inte kunnat utnyttjas ordentligt tidigare. Anledningen till att AI har fått stort genomslag inom en mängd applikationer i olika branscher först idag är främst för att det nu finns kraftfullare datorer, större tillgång på information men också att det finns en rad färdiga modeller och resurspaket som är enkla att tillämpa och använda.

Intelligens definieras som förmågan att uppfatta, lära och därefter agera [8]. AI är i grunden en teknik som gör det möjligt för datorer att härma intelligent beteende

och lösa avancerade problem [9]. Maskininlärning ses som en underkategori till AI [9] och är ett forskningsområde som bygger på att ge datorer möjlighet att lära utan att explicit programmera dem [10]. Det är ett sätt att bygga algoritmer så att de är självlärande, med andra ord algoritmer för dataanalys som bygger och justerar modeller efter den tillgängliga indatan [7]. Detta kraftfulla verktyg möjliggör en automatiserad analys av problem och används särskilt för att hantera och dra slutsatser ifrån stora mängder data [10].

Lastprognostisering handlar om att göra prognoser av framtida last på elnätet. För en konsument motsvarar denna last konsumentens elförbrukning. I denna rapport används elförbrukning och elektrisk last synonymt. För att lastprognostisering ska kunna användas som hjälpmedel vid beslut om energihantering behöver prognosen ha hög precision. För att förbättra prognosprecision har studier av lastprognostisering med hjälp av AI genomförts. Dessa studier har gjorts nästan uteslutande med en upplösning på en timme eller längre [11]. Däremot finns intresse i att generera en prognos med högre upplösning för att möjliggöra smartare beslut och planering för elförbrukning och -försäljning [12].

1.1 Syfte

Syftet med detta projekt är därför att utveckla och jämföra AI-modeller i programmeringsspråket Python för kortsiktig och högupplöst lastprognostisering.

1.2 Problemformulering

Eftersom forskarsamfundet har uttryckt behov av lastprognostiseringsverktyg med högre tidsupplösning ämnar projektet skapa och jämföra lastprognostiseringsverktyg med 15 minuters upplösning. Prognostisering med hög upplösning ger på lång sikt stor osäkerhet men behovet för hög upplösning minskar också vid prognostisering långt fram i tiden [12]. Med denna motivering begränsas tidshorisonten för prognoserna i projektet till 24 timmar.

Tre ofta använda maskininlärningsmetoder för lastprognostisering är linjär regression, bakåtpropagerande artificiellt neoronnät (eng. Artificial Neural Network - ANN) och återkommande neuronnät (eng. Recurrent Neural Networks - RNN) med långt korttidsminne (eng. Long Short Term Memory - LSTM) [13]–[15]. Eftersom metoderna är vanligt förekommande och för att bibehålla en rimlig omfattning på projektet avses endast dessa tre metoder jämföras. För att uppfylla projektets syfte ska följande frågeställningar besvaras:

- Hur kan lastprognostiseringsverktyg baserade på de tre metoderna utvecklas och optimeras i Python?
- Vilken av de tre metoderna producerar bäst lastprognostisering på mindre flerbostadshus med 15 minuters upplösning och 24 timmars tidshorisont? Med bra prognostisering menas litet och väldefinierat fel och med väldefinierat menas att felet kan approximeras väl med en statistisk fördelning som en Gaussisk fördelning.

1.3 Avgränsningar

Maximalt tio veckor av projekttiden kommer vigas åt att bygga och optimera valda modeller. En begränsning i tid har valts då det finns många tänkbara konfigurationer och parametrar att ändra och detta annars kan göras närmast oändligt.

Datamängd och datorkraft för modellerna ska vara av den storlek att de ska kunna köras på en modern PC. Detta för att verktyget ska kunna användas i praktiken utan att kräva dyr specialutrustning.

Använd data kommer begränsas till lastdata från HSB living lab samt publik väderdata från SMHI. Detta för att få verktyget generaliserbart då historisk lastdata och publik väderdata bör finnas tillgänglig för alla som önskar använda verktyget.

1.4 Etiska och samhälleliga aspekter

Under detta arbete förekommer en rad etiska och/eller samhälleliga aspekter som bör beaktas. De etiska aspekter som anses vara relevanta för detta arbete är framför allt hållbarhet, vilken nytta projektet bidrar eller skulle kunna bidra till, samt integritet.

Utomvetenskapligt kan konstateras att projektet är ett arbete som i grunden bidrar till lärdom och förståelse för de involverade studenterna inom projektgruppen. Ett välfungerande verktyg för lastprognostisering tros kunna medverka till att olika aktörer och brukare, mer precist kan analysera och uppskatta hur förbrukning av el kommer att se ut i närtiden. Dessa brukare kan vara allt ifrån privatpersoner i mindre hushåll till elbolag i större system. Det innebär också att ett elbolag med tillgång till sådan utrustning skulle kunna missbruka den ur ekonomiskt intresse, exempelvis genom att anpassa elpriser eller begränsa tillgången efter behov i syfte att göra större vinster. I nuläget existerar redan många studier som gjorts för lastprognostisering över större områden [16]–[18] och bidrar med en möjlighet för elleverantörer att justera priset efter behov. Eftersom detta projekt fokuserar på att prognostisera på individ-/konsumentnivå antas att det inte kommer att bidra till någon sådan förändring.

I fallet att den färdiga produkten skulle prognostisera lasten mycket precist och va-

ra brett tillämpningsbar hade utfallet gjort förnyelsebara, intermittenta energikällor mer attraktiva för både privat bruk och för större aktörer samt bidragit till en större implementering av AI-teknik i elnäten. Betraktat från ett makroskopiskt perspektiv talar sådan teknik för att stora mängder energi kan sparas, eftersom elproduktionen i ett idealfall alltid kan satisfiera elbehovet som prognostiseras utan att överflödig, svårlagrad el behöver genereras [19].

Beroende på prestanda, komplexitet, upplösning och andra faktorer i algoritmerna krävs dock en potentiellt stor mängd användardata [20]. Ur ett integritetsperspektiv är det därför viktigt att utformningen av verktyget sker på ett sätt som inte kränker någon juridisk gräns och att användaren är väl medveten om hur verktyget fungerar samt vilken data som används. För att kringgå sådana integritetsproblem används endast publik väderdata från SMHI, samt lastdata från forskningsboendet HSB living lab, vars tillämpning i projektet är godkänt av residenterna.

Artificiell intelligens är en teknik som uppnår sin träffsäkerhet genom att träna på historisk data. Vad som är viktigt att belysa inom detta är huruvida den prognostiserade användningen av energi bör vara den som eftersträvas. Ur ett hållbarhetsperspektiv är det relevant att eftersträva en minskning av energiförbrukningen i samhället. Ett verktyg som prognostiserar baserat på historisk data kan ge incitament att tro att den prognostiserade energianvändningen är den rätta. Därför är det viktigt att ta i beaktning att enbart för att prognosen anger en viss energianvändning, betyder det inte att det är vad som ska strävas mot.

2 Teoretisk referensram

Under projektet utvecklas tre olika typer av AI-modeller som ska utvärderas, jämföras och slutligen resultera i ett färdigt prognostiseringsverktyg. De tre modellerna är linjär regression (LR), artificiella neuronnät (eng. Artificial Neural Networks - ANN) med bakåtpropagering (eng. backpropagation) samt återkommande neuronnät (eng. Recurrent Neural Networks - RNN) med långt korttidsminne (eng. Long Short-Term Memory - LSTM) och förklaras djupare i delkapitel 2.1-2.3. Samtliga modellers inlärningsprocesser är av typen övervakad inlärning, vilket innebär att indatan som AI-modellerna hanterar är klassificerad och parad med en avsedd utdata [21]. Resultatet blir att algoritmerna anpassar en funktion mellan indata och utdata som kan hantera många parametrar.

2.1 Linjär regression

Regressionsanalys är en statistisk metod för att undersöka och modellera relationer mellan variabler [22]. Inom maskininlärning används regression för att utifrån ett antal invariabler, regressorer, förutspå ett numeriskt värde, utvärdet [23]. Modellen består av en funktion f, sådan att

$$f : \mathbb{R}^n \rightarrow \mathbb{R}. \tag{2.1}$$

Linjär regression är en grundläggande modell inom datavetenskap (eng. Data Science) och analys [23]. I linjär regression kallas fallet då $n = 1$ i ekvation (2.1) enkel linjär regression och beskrivs enligt

$$y = \beta_0 + \beta_1 x + \epsilon. \tag{2.2}$$

Enkel linjär regression innebär passning av ekvation (2.2) med parametrarna β_0, β_1, till en regressor x och motsvarande utvärde y så att felet ϵ minimeras. Multipel linjär regression är en generalisering av ekvation (2.2) där y relateras till flera regressorer, x_1, x_2, ..., x_k,

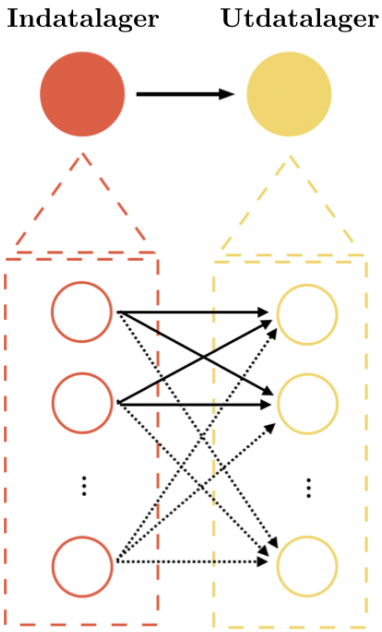
$$\text{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k + \epsilon, \tag{2.3}$$

där ekvation (2.3) är linjär i parametrarna β_0, β_1, β_2, ... , β_k [24]. Parametrarna kan ses som vikter, vilka avgör hur mycket varje regressor påverkar utvärdet. I linjär regression anpassas vikterna till ekvationen så att felet ϵ, minimeras.

Vidare kan ekvationen (2.3) generaliseras ytterligare till multivariat linjär regression, där $\mathbf{Y}$, $\mathbf{X}$, $\boldsymbol{\beta}$ och $\boldsymbol{\epsilon}$ är matriser. Detta ger

$$\mathbf{Y}_{m\times q} = \mathbf{X}_{m\times n}\boldsymbol{\beta}_{n\times q} + \boldsymbol{\epsilon}_{m\times q}. \tag{2.4}$$

Denna generalisering är särskilt relevant i den här rapporten då värden vid flertalet tidpunkter framåt sökes. Med $m = 1$ fås den ekvation som löses för vår modell. Då gäller att q är antalet utvariabler och n är antalet invariabler. Den grafiska motsvarigheten till den modell som beskrivs i ekvation (2.4) med $m = 1$ ges i figur 2.1.



Figur 2.1: Konceptuell figur över en linjär regressionmodell med flera indata- och utdatapunkter.

Funktionen som minimeras för att minska felet $\boldsymbol{\epsilon}$ kallas kostnadsfunktionen C. Den minimeras genom att justera parametrarna i β. För multivariat linjär regression och fallet $m = 1$ gäller typiskt kostnadsfunktionen

$$C = \sum_{i=1}^{q}\left(y_i - \sum_{j=0}^{n} x_j \times \beta_{ji}\right)^2, \tag{2.5}$$

där q och n anger antalet ut- respektive invariabler. Användning av kostnadsfunktionen i (2.5) kallas också minsta kvadratmetoden. Den klassiska linjära regressionsmodellen kan utvecklas genom att justera hur parametrarna β_0, ..., β_n optimeras. Felet ϵ minimeras då med andra funktioner än den klassiska minsta kvadratmetoden.

2.1.1 Utveckling av klassisk linjär regression

I Ridge-regression utökas kostnadsfunktionen i (2.5) med ett straff som ges av summan av koefficienterna β_{ji} i kvadrat. Med samma beteckningar som i (2.5) ger detta

$$C = \sum_{i=1}^{q} \left(\left(y_i - \sum_{j=0}^{n} x_j \times \beta_{ji} \right)^2 + \lambda \sum_{j=0}^{n} \beta_{ji}^2 \right). \tag{2.6}$$

Ekvation (2.6) innebär att höga vikter β_{ji}, straffas. Alltså stora λ ger låga β_{ji}.

Lasso-regression (eng. Least Absolute Selection and Shrinkage Operator) verkar också för att minska koefficienterna, med den viktiga skillnaden att summan av absolutbeloppet av koefficienterna straffas istället för summan av kvadraten av koefficienterna. I lasso-regression ges kosnadsfunktionen istället av

$$C = \sum_{i=1}^{q} \left(\left(y_i - \sum_{j=0}^{n} x_j \times \beta_{ji} \right)^2 + \lambda \sum_{j=0}^{n} |\beta_{ji}| \right). \tag{2.7}$$

Detta har liknande effekter som ridge-regression, det vill säga motverkar överträning av modellen genom att straffa höga vikter. Eftersom det är absolutbeloppet, inte kvadraten av vikten som straffas blir resultatet något annorlunda. Istället för att vikterna går mot noll då λ går mot oändligheten kommer vikterna bli exakt noll.

För både ridge- och lasso-regression är det storleken på λ som avgör hur mycket höga parametervärden/vikter straffas i modellen. Då λ är noll reduceras både kostnadsfunktionen för ridge-regression (2.6) och lasso-regression (2.7) till kostnadsfunktionen för klassisk linjär regression (2.5).

Principalkomponentregression (PCR) bygger på principalkomponentanalys vilken är en linjär ortogonal transform av indatan som gör att den transformerade datans dimensioner är ortogonala, det vill säga att dimensionerna är oberoende och inte har någon korrelation till varandra. Detta görs genom att hitta hyperplan i indatan med maximal varians till utdatan. Typiskt rensas också delar i datan med låg korrelation till utdatan bort. Vidare utförs multipel linjär regression på vanligt vis men på ett indataset som bör ha tydligare linjär korrelation med utdatan samt saknar multikollinearitet.

Partiell minsta kvadrat regression (PLSR) liknar till viss del PCR men istället för att hitta hyperplan med maximal varians projicerar PLSR indatan **X** på ett hyperplan **T** och utdatan **Y** på ett hyperplan **U** så att **T** förklarar **U** väl. Regression mellan **T** och **U** utförs sedan istället för regression mellan **X** och **Y**. Representationen förklaras typiskt genom matrismultiplikationerna

$$\begin{aligned} \mathbf{X} &= \mathbf{T}\mathbf{P}^T + \mathbf{E} \\ \mathbf{Y} &= \mathbf{U}\mathbf{Q}^T + \mathbf{F}, \end{aligned} \tag{2.8}$$

där **P** och **Q** är ortogonala matriser som relaterar den verkliga in- och utdatan **X** och **Y** till deras representationer **T** och **U**. Vidare är **E** och **F** feltermer. Styrkan i PLSR är att kunna utnyttja information i både in- och utdata och på så sätt göra en smart transformering för att sedan relatera dem genom regression [25].

2.1.2 Tidigare resultat

Tidigare forskning har undersökt linjära regressionsmodeller för lastprognostisering med en upplösning på en timme och en horisont på 24 timmar [15], [26], [27], förutom i [28] där en horisont på ett till sju dygn undersökts.

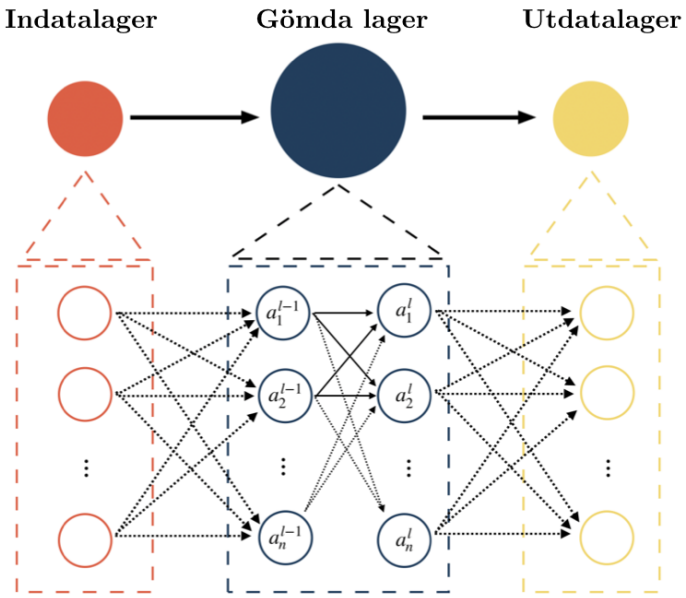
Dessa studier har redovisat prognoser med MAPE-fel på 3,52-4,34% [15], 1,00-2,63% [26], 4,98% [28] och 5,20-6,10% [27]. Resultaten från de olika tidigare studierna skiljer smått vilket kan bero på flera olika faktorer, som applikation, mätområde, upplösning och tidshorisont. En kort sammanfattning av MAPE-fel, upplösning, tidshorisont och applikation hittas i tabell 2.1.

Tabell 2.1: Sammanfattning av tidigare studiers resultat med linjär regressionsmodell som verktyg för prognostiseringen.

MAPE [%]	Upplösn.	Tidshorisont	Applikation	Ref.
3,52-4,34	1 h	24 h	Provins i indonesien	[15]
1,00-2,63	1 h	24 h	Polens elsystem	[26]
5,20-6,10	1 h	24 h	782 hushåll	[27]
4,98	1 h	1-7 dygn	Region i USA	[28]

2.2 Bakåtpropagerande artificiellt neuronnätverk (ANN)

En mer komplex modell är artificiellt neuronnätverk (ANN). ANN är en väldigt flexibel maskininlärningsmodell och används inom många olika områden, bland annat inom lastprognistisering [16], [17], [29]–[31].

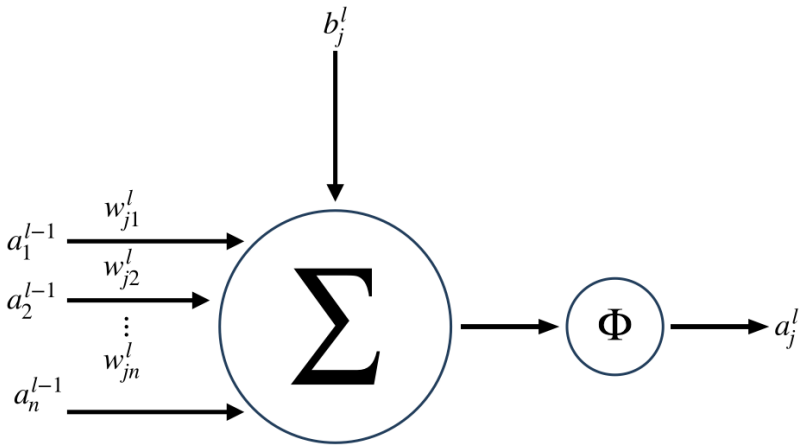


Figur 2.2: Illustration av ett artificiellt neuronnäts uppbyggnad, där a_n^l är nod n i lager l. Cirklar representerar noder och pilar representerar vikter mellan noder och lager.

ANN är konstruerade på ett sätt som är tänkt att imitera människans hjärna, med olika neuroner (noder) som har olika starka kopplingar till varandra. Noder är grunden för hela neuronnätverket och i figur 2.2 ses en överblick över hur ett ANN kan se ut. I figuren representerar cirklar noder, pilar representerar vikter mellan noder och färgerna indikerar de olika delar som nätverket består av.

En nods värde beror på föregående noder, ett bias samt en aktiveringsfunktion, där biaset kan ses som ett justeringsvärde eller startvärde, likt β_0-värdet i linjära regressionens ekvationer (2.2), (2.3). En aktiveringsfunktion matas med den viktade summan av alla föregående noder och biaset. Det utvärde som fås av aktiveringsfunktionen är nodens värde och kallas nodens aktivering. I figur 2.3 ses en illustration av denna process, där a_n^l är nod n i lager l, w_{jn}^l är vikten mellan nod n i lager $l-1$ och

nod j i lager l, b_j^l är biaset för nod j i lager l och där Φ representerar aktiveringsfunktionen för nod j.



Figur 2.3: Illustration av hur en nod är uppbyggd, där a_n^l är nod n i lager l, w_{jn}^l är vikten mellan nod n i lager $l-1$ och nod j i lager l, b_j^l är bias för nod j i lager l. Den inringade Φ-symbolen representerar här en godtycklig aktiveringsfunktionen för nod j.

Från nodens uppbyggnad och nätverkets struktur kan utdatan $\mathbf{a}^L$ för ett nätverk med L lager beräknas för indata $\mathbf{x}$. Enligt samma notationer som tidigare kan en nods aktivering matematiskt ges enligt

$$a_j^l = \Phi\left(\sum_k w_{jk}^l \cdot a_k^{l-1} + b_j^l\right). \tag{2.9}$$

Genom vektorisering av (2.9) fås (2.10), där $\mathbf{a}^l \in \mathbb{R}^{1\times n}$ innehåller aktiveringsvärde för alla n noder i lager l, $\mathbf{W}^l \in \mathbb{R}^{n\times k}$ innehåller vikter mellan lager l med n noder och lager $l-1$ med k noder och $\mathbf{b}^l \in \mathbb{R}^{1\times n}$ innehåller bias för alla n noder i lager l.

$$\mathbf{a}^l = \Phi(\mathbf{W}^l \cdot \mathbf{a}^{l-1} + \mathbf{b}^l) \tag{2.10}$$

Från (2.10) kan utdatan $\mathbf{a}^L$ beräknas genom att sätta indata $\mathbf{x}$ till $\mathbf{a}^1$ och iterativt applicera ekvation (2.10) för varje lager tills slutet nås. Utdatan $\mathbf{a}^L$ som funktion av indata $\mathbf{x}$ och parametrar $\mathbf{W}$ och $\mathbf{b}$ ges då av

$$\mathbf{a}^L = \Phi(\mathbf{W}^L\Phi(\mathbf{W}^{L-1}\Phi(...(\mathbf{W}^2\mathbf{a}^1 + \mathbf{b}^2)...) + \mathbf{b}^{L-1}) + \mathbf{b}^L). \tag{2.11}$$

Ett bakåtpropagerande ANN lär sig genom att träna på indata som har associerad utdata, alltså ett värde på indatan svarar mot ett värde på utdatan. Efter jämförelse mellan nätverkets utdata och förväntad utdata kan vikter och bias uppdateras för att minimera felet [32]. Detta görs med hjälp av en kostnadsfunktion C, vilken är en funktion av nätverkets utdata och förväntad utdata och beskrivs av

$$C = f(\mathbf{a}^L, \mathbf{y}), \tag{2.12}$$

där $\mathbf{a}^L$ är nodernas värden i sista lagret och $\mathbf{y}$ är önskad utdata. Den mest använda funktionen f i regressionsproblem är genomsnittligt kvadrerat fel (eng. Mean Squared Error - MSE) [32].

För att få nätverket att prestera optimalt, gäller det att minimera kostnadsfunktionen. Detta görs genom att stegvis ändra värdena på vikter och bias. Algoritmen för detta kallas bakåtpropagering och tar fram gradienten, ∇C för kostnadsfunktionen i aktuell punkt. Med hjälp av denna finns flertalet optimeringsmetoder för hur vikter och bias ska uppdateras för att minimera kostnadsfunktionen [32]. Dessa bygger i regel på stokastisk gradientminskning, där optimeringsmetoden uppskattar gradienten genom att använda sig av en mindre serie (eng. batch) data i taget för att minska beräkningstyngden [32].

Optimeringsfunktioner innehåller i regel andra parametrar där den med störst betydelse är inlärningshastighet (eng. learning rate) som styr hur stora förändringar som görs i varje iteration [33]. En för låg inlärningshastighet orsakar en onödigt långsam träningsprocess där det även finns risk att fastna i lokala minimum. En för hög inlärningshastighet medför en snabbare träningsprocess men kan istället hindra vikterna från att konvergera [33]. Val av optimal inlärningshastighet beror mycket på problemet problemet som ska lösas.

Nätverket tränas iterativt genom att mata det med serier av indata, beräkna gradienten med bakåtpropagering och uppdatera vikter och bias för att minska kostnadsfunktionen, vilket upprepas tills att vikter och bias har konvergerat mot optimala värden och träningen är klar. För att bedöma detta används ofta separat valideringsdata för att undvika att nätverket överanpassas till träningsdatan [32].

2.2.1 Tidigare resultat

Några tidigare studier som genomförts gällande lastprognostisering med hjälp av ANN presenteras i tabell 2.2. I studier på en 24-timmarshorisont, presenteras MAPE-värden på 15,32% [29], 7,19% [30], 14,50% [31] och på en 48-timmarshorisont presenteras värdet 20,44% [29]. Det är många faktorer som påverkar resultaten av de olika studierna inklusive applikation, mätområde, upplösning och horisont. Jämförelser mellan studier blir således svåra. En sammanfattning av dessa faktorer för studierna hittas i tabell 2.2.

Tabell 2.2: Sammanfattning av tidigare studiers resultat med ANN som verktyg för prognostiseringen.

MAPE [%]	Upplösn.	Tidshorisont	Applikation	Referens
15,32	1 h	24 h	Byggnadskomplex	[29]
20,44	1 h	48 h	Byggnadskomplex	[29]
7,19	1 h	24 h	Campusbyggnad	[30]
14,50	1 h	24 h	69 hushåll	[31]

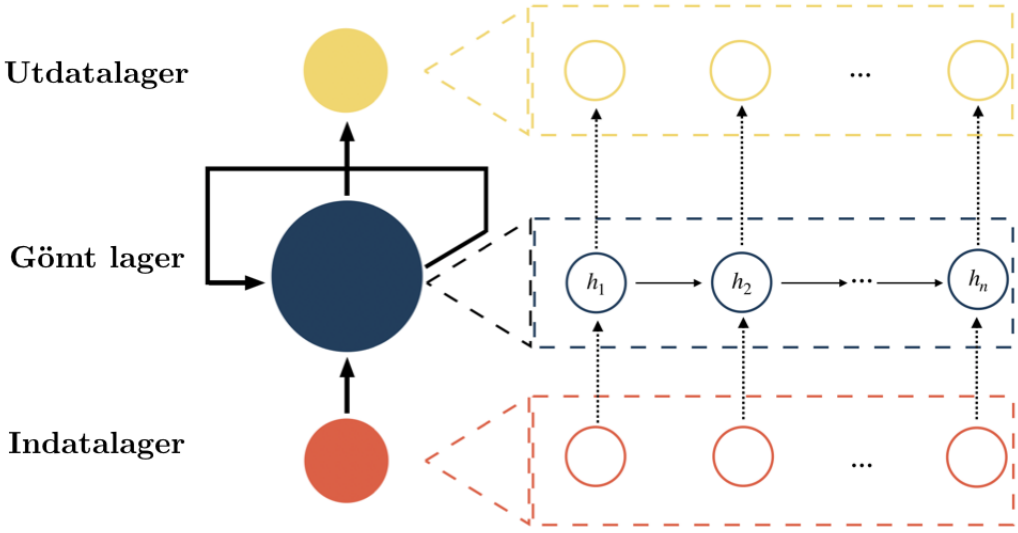
2.3 Återkommande neuronnät med långt korttidsminne

En annan klass av AI-modeller är återkommande neuronnät (RNN). I följande kapitel introduceras ett generellt koncept för RNN och två vanligt förekommande problem med försvinnande och exploderande gradienter. Dessutom presenteras ett långt korttidsminne (LSTM) som är en typ av RNN och till stor del åtgärdar dessa problem.

2.3.1 Generellt om återkommande neuronnät (RNN)

Ett RNN är en form av neuronnät som utmärker sig genom dess förmåga att skapa ett slags minne. Nätverket åstadkommer detta genom att information inte bara passerar från lager till lager, utan även mellan noder inom samma lager. Denna teknik är fördelaktig vid behandling av olika stora sekventiella dataset och gör RNN till en robust modell för bland annat igenkänning och prognostisering inom tal, skrift eller andra applikationer där datan är sekventiellt strukturerad [34].

I figur 2.4 presenteras en konceptuell skiss av ett RNN. Designen för ett RNN liknar till stor del designen av ett ANN. Generellt består modellen av en inmatning som viktas och passerar till en nod i ett gömt lager och en utdata. I det gömda lagret itereras noden över varje steg i sekvensen eller tidsserien. Ur ett konceptuellt perspektiv kan detta föreställas i form av att ett RNN lägger till en slinga i noden som återkopplas för varje inmatning. Den slingade noden kallas även för cell eller återkommande cell (eng. recurrent cell). Med denna återkoppling genereras ett korttidsminne mellan varje tidssteg som möjliggör att modellen lär sig kortsiktiga samband oberoende av sekvenslängden. Denna form av minne kallas för dolt tillstånd (eng. hidden state) och noteras h. Det bör observeras att det kan förekomma ett godtyckligt antal gömda lager i ett RNN men att beskrivningen i detta projekt av förenklingsskäl endast behandlar strukturer med ett gömt lager. Även kopplingar mellan cellerna kan i praktiken göras på olika sätt, både mellan celler i olika lager och mellan celler i samma lager. Beskrivningen i detta kapitel är dock begränsad till grundstrukturen som syns i figuren.



Figur 2.4: Illustration över dataflöde i RNN där h_1 är dolt tillstånd för tidssteg 1, h_2 är dolt tillstånd för tidssteg 2 och h_n är dolt tillstånd för tidssteg n. Icke ifyllda cirklar representerar noder och pilar representerar vikter mellan noder och lager.

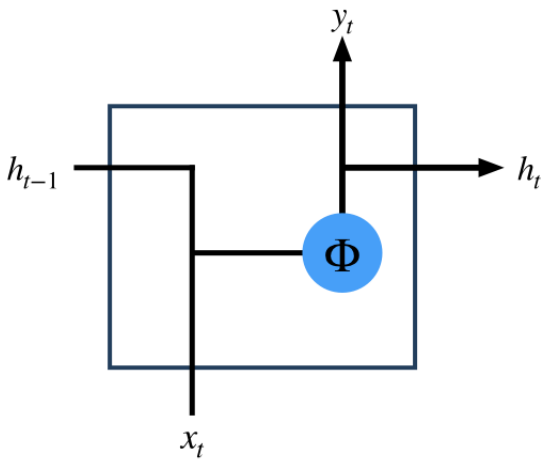
RNN-cellen i tidssteg t visas i figur 2.5. Grunden för varje cells bearbetning av indata bygger på ekvation (2.9) men med en modifikation för att den sekventiella funktionaliteten och tidigare indata ska tas i beaktning. Det dolda tillståndet som passerar framåt mellan cellerna tecknas

$$h_t = \Phi_{\text{tanh}} \left(w_{xh} x_t + w_{hh} h_{t-1} - b_h \right), \tag{2.13}$$

där Φ_{tanh} är aktiveringsfunktionen tangens hyperbolicus (tanh), x_t och h_{t-1} är indata ifrån tidigare lager respektive dolt tillstånd ifrån tidigare cell i samma lager, w_{xh} och w_{hh} är de tillhörande vikterna och b_h är ett bias. Vikterna i ett RNN varierar inte inom ett lager och är därför oberoende av t. På liknande sätt kan utdatan till nod t i nästa lager tecknas

$$y_t = \Phi_y \left(w_{yh} h_t - b_y \right), \tag{2.14}$$

där Φ_y är en godtycklig aktiveringsfunktion, w_{yh} är vikten mellan cell t och nästa lager och b_y är tillhörande bias.



Figur 2.5: Skiss över RNN-cell där h_{t-1} är det dolda tillståndet ifrån tidssteg $t-1$, x_t är indata till cellen, h_t är dolt tillstånd och y_t är utdata ifrån cellen vid tidssteg t.

I ekvation (2.13) och (2.14) kan noteras att tidigare vikt och inmatning tas i beaktning innan aktiveringsfunktionen implementeras. Valet av antalet utdatapunkter y_t varierar beroende på vilken typ av prognos som ska genereras. I den konceptuella modellen i figur 2.4 åskådliggörs en version där varje cell ger upphov till var sin utdata. Principiellt kan detta modifieras för att anpassa eller optimera modellen efter prognosens behov. Justering av antalet gömda lager samt hur cellerna är kopplade till varandra är också metoder för att anpassa och optimera modellen.

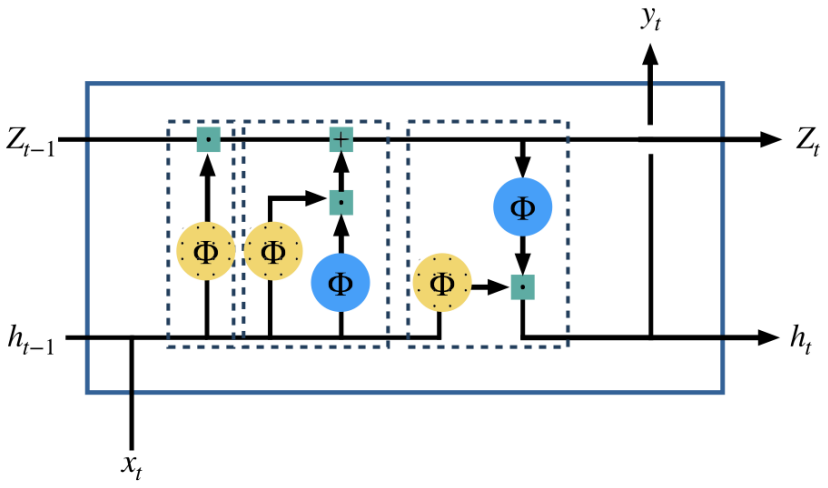
2.3.2 Försvinnande och exploderande gradienter

RNN använder sig likt andra neuronnät av bakåtpropagering men med en modifikation som kallas bakåtpropagering genom tid (eng. BackPropagation Through Time - BPTT) vid träning och inlärning. Vid BPTT i nätverksstrukturer med långa sekvenser av indata kan det uppstå problem med försvinnande och exploderande gradienter [35]. Inlärningsprocessen realiseras genom att algoritmen modifierar vikterna w_{xh}, w_{hh} och w_{yh} med hänsyn till summan av felen i varje utdata. Uppdateringen av vikterna kan skrivas som derivatan (gradienten) av felet med avseende på vikterna. Dessa utvecklas sedan med hjälp av kedjeregeln över samtliga lager och tidssteg. En struktur med många lager och sekvenser med många tidssteg kommer därför att innehålla många upprepade multiplikationer. Problemet uppstår när faktorerna

är små och multiplikationen konvergerar mot noll. I praktiken medför detta att den dynamiska uppdateringen av tidiga cellers vikter, det vill säga inlärningen, sker väldigt långsamt eller i värsta fall upphör genom BPTT. På liknande sätt kan det uppstå problem där vikterna går mot höga värden vilket kallas för exploderande gradientproblem, vilket leder till större svängningar och förhindrar konvergens. Båda utsagor ger negativa konsekvenser på träningen av modellen.

2.3.3 Långt korttidsminne (LSTM)

För att till stor del undvika problemet kring försvinnande gradienter vid träning av långa sekvenser kan RNN-cellerna ersättas med LSTM-celler [36]. En LSTM-cell fungerar likt en RNN-cell men är mer komplext strukturerad i syfte att etablera ett selektivt informationsflöde. Detta möjliggörs med hjälp av fler aktiveringsfunktioner och införandet av ett så kallat celltillstånd (eng. cell state) Z som varierar med tidssteg t. Celltillståndet är en avgörande del av det som konceptuellt kan beskrivas som modellens långtidsminne. I figur 2.6 presenteras en LSTM-cell med två olika typer av aktiveringsfunktioner där gula cirklar med prickar representerar en sigmoidfunktion och blåa en tanh-funktion.



Figur 2.6: LSTM-cell där h_{t-1} är dolt tillstånd och Z_{t-1} är celltillstånd från tidssteg $t-1$. För tidssteg t finns indatan x_t, utdata y_t samt det nya uppdaterade celltillståndet Z_t och dolt tillstånd h_t. Markerat med tre streckade rektanglar ifrån vänster är LSTM-cellens portar (glömmaport, uppdateraport och utdataport). Gul och prickig cirkel representerar aktiveringsfunktionen sigmoid och blå cirkel aktiveringsfunktionen tanh.

Vid jämförelse mot grundvariant i figur 2.5 kan skillnader i indata till cellen observeras. Här representerar Z_{t-1} celltillståndet från föregående cell som också kan betraktas som ett transportband för information mellan cellerna i ett lager. Informationshanteringen i LSTM-cellen kan grupperas efter tre olika portar. De tre portarnas funktioner är:

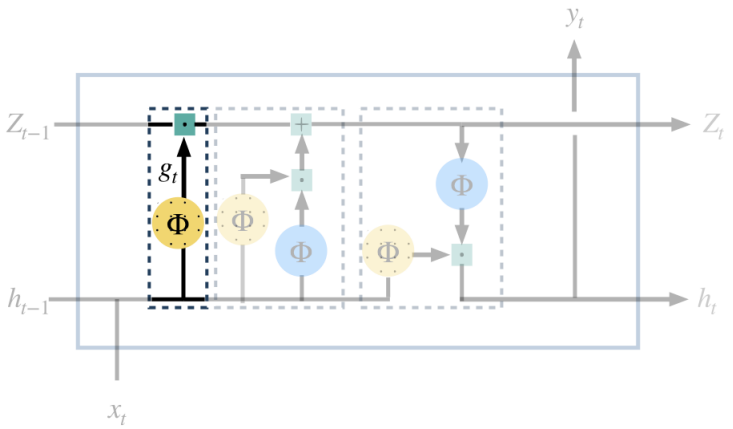
1. Glömmaport.
2. Uppdateraport.
3. Utdataport.

I figur 2.6 visas portarna i samma ordning från vänster till höger i streckade rutor. Med LSTM-cellens uppbyggnad av portar och införandet av celltillstånd kan gradientens värde under träningen bättre kontrolleras och därigenom minska risken för försvinnande och exploderande gradienter.

I figur 2.7 är glömmaporten framhävd. I denna port normeras indata med en sigmoid-funktion för att få ett värde mellan noll och ett. Detta medför att information som inte ska sparas kommer att få ett värde nära noll, respektive ett för det omvända fallet. Utdatan från glömmaporten kan beräknas som

$$g_t = \Phi_{\text{sigmoid}}(x_t w_{xg} + h_{t-1} w_{hg} + b_g), \tag{2.15}$$

där w_{xg} och w_{hg} är tillhörande vikter för indatan x_t respektive h_{t-1}, b_g är aktuell bias och Φ_{sigmoid} är den normerande sigmoid-funktionen.



Figur 2.7: Framhävt i figuren är LSTM-cellens glömmaport där g_t representerar den information som kommer ut ifrån sigmoid-aktiveringsfunktionen här representerad av gul och prickig cirkel.

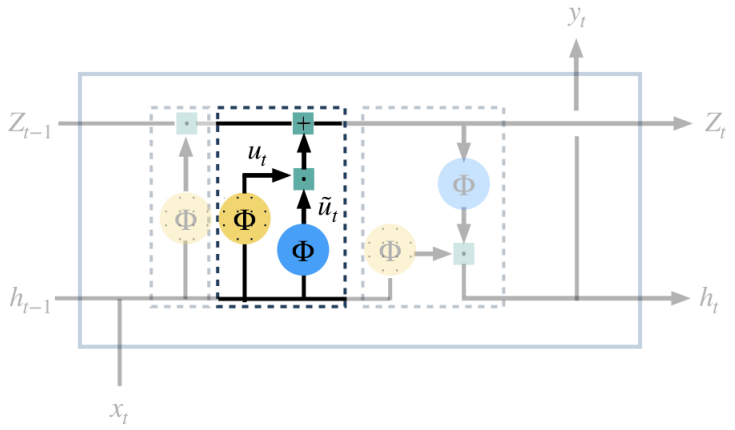
I figur 2.8 är uppdateringsporten framhävd. Uppdateringsporten kan i sin tur delas upp i två delar. Den första avgör vilka värden som ska tas i beaktning via en sigmoid-funktion i den gula cirkeln med prickar. Utdatan från denna port beräknas som

$$u_t = \Phi_{\text{sigmoid}}(x_t w_{xu} + h_{t-1} w_{hu} + b_u), \tag{2.16}$$

där w_{xu} och w_{hu} är tillhörande vikter för indatan x_t respektive h_{t-1}, och b_u är aktuell bias. Den andra delen är den blåa cirkeln som normerar värdena i intervallet (-1, 1) enligt normeringsfunktionen tanh, för att medföra en skalning av hur mycket informationen ska betyda vilket representeras av $\tilde{u}_t$. Utdatan kan tecknas

$$\tilde{u}_t = \Phi_{\text{tanh}}(x_t w_{x\tilde{u}} + h_{t-1} w_{h\tilde{u}} + b_{\tilde{u}}), \tag{2.17}$$

där $w_{x\tilde{u}}$ och $w_{h\tilde{u}}$ är tillhörande vikter för indatan x_t respektive h_{t-1}, $b_{\tilde{u}}$ är aktuell bias och Φ_{tanh} är den normerande tanh-funktionen.



Figur 2.8: Framhävt i figuren är LSTM-cellens uppdateraport där u_t är den information som kommer ut ifrån sigmoid aktiveringsfunktionen här representerat av gul och prickig cirkel och $\tilde{u}_t$ är den information som kommer ut ifrån tanh-aktiveringsfunktionen här representerad av en blå cirkel.

Genom kombination av resultatet ur ekvationerna (2.15)-(2.17) kan det nya celltillståndet Z_t erhållas enligt

$$Z_t = g_t \cdot Z_{t-1} + \tilde{u}_t \cdot u_t, \tag{2.18}$$

där Z_{t-1} är celltillståndet i föregående tidssteg.

I figur 2.9 är utdataporten framhävd. I utdataporten normeras information ifrån cell-tillståndet Z_t med tanh som aktiveringsfunktion enligt

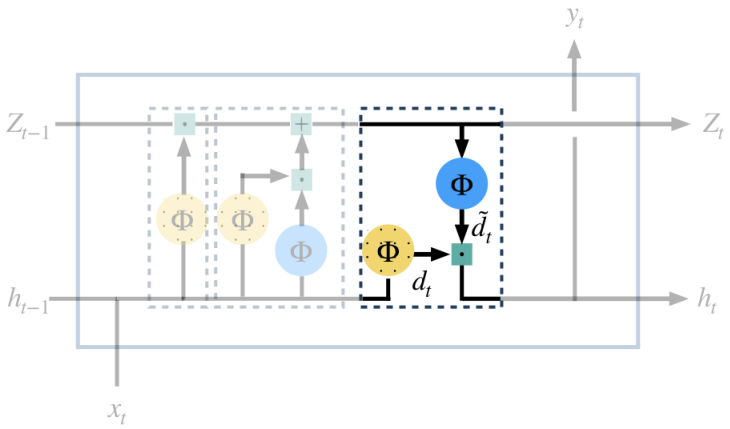
$$\tilde{d}_t = \Phi_{\text{tanh}}(Z_t \cdot w_{\tilde{d}} + b_{\tilde{d}}), \tag{2.19}$$

där $w_{\tilde{d}}$ och $b_{\tilde{d}}$ är tillhörande vikt respektive aktuell bias. Dessutom bestäms vilken information som passeras genom sigmoid-aktiveringsfunktionens normering av den ursprungliga indatan enligt

$$d_t = \Phi_{\text{sigmoid}}(x_t w_{xd} + h_{t-1} w_{hd} + b_d), \tag{2.20}$$

där w_{xd} och w_{hd} är tillhörande vikter och b_d är aktuell bias. Punktvis multiplikation mellan $\tilde{d}_t$ och d_t resulterar i att det dolda tillståndet samt utdatan till cell t i nästa lager slutligen kan beräknas genom

$$h_t = \tilde{d}_t \cdot d_t. \tag{2.21}$$



Figur 2.9: Framhävt i figuren är LSTM-cellens utdataport där d_t är den information som kommer ut ifrån sigmoid-aktiveringsfunktionen här representerad av gul och prickig cirkel och $\tilde{d}_t$ är den information som kommer ut ifrån tanh-aktiveringsfunktionen här representerat av en blå cirkel.

LSTM-strukturen som presenterats i figur 2.6 och ekvationerna (2.15)-(2.21) är en variant av strukturell uppbyggnad. LSTM har stora möjligheter att anpassas mot variande problem [37].

2.3.4 Tidigare resultat

LSTM-modeller har i tidigare forskning inom liknande områden uppnått högre precision än många andra etablerade modeller [38]–[41]. I en studie redogörs för att LSTM-modellen är bättre applicerbar för kortare tidshorisonter jämfört med andra AI-modeller [40]. I tabell 2.3 presenteras resultat ifrån tidigare användning av LSTM-modeller inom lastprognostisering för att skapa en uppfattning om storleksordningen av MAPE. Applikationen för implementeringen av LSTM-modell varierar både med avseende på lastscenario, tidshorisont och prognosens upplösning.

Tabell 2.3: Sammanfattning av tidigare studiers resultat med LSTM som verktyg för prognostiseringen.

MAPE [%]	Upplösn.	Tidshorisont	Applikation	Ref.
8,58	1 h	24 h	69st hushåll	[31]
6,54	1 h	1 h	ISO New England	[40]
5,35	15 m	24 h	Skolbyggnad	[42]
6,45-14,01	10 m	24 h	Byggnad	[43]

3 Metod

För utveckling av lastprognostiseringsprogrammen användes programmeringsspråket Python. Anledningar till att just Python valdes var dels den goda tillgängligheten till öppna och väldokumenterade programbibliotek för AI-algoritmer, god läslighet av kod, samt att språket är välanvänt inom tidigare forskning och generellt bland utvecklare [44]. I tabell 3.1 presenteras de programbibliotek som användes.

Tabell 3.1: Använda öppna Python programbibliotek vid programmeringen.

Programbibliotek	Scikit-learn	TensorFlow	Matplotlib	pandas	Keras	NumPy
Version	0.24.1	2.4.1	3.3.2	1.2.1	2.4.3	1.19.5

Versionen för Python som användes är 3.8.5. För att kunna återskapa projektet och bidra till vidare forskning inom ämnet gjordes programkoden för modellerna tillgänglig på GitHub:

`https://github.com/ellinorl/power_consumption_forecasting`

3.1 Indata

Utvecklandet och analysen av modellerna utfördes med hjälp av lastdata från HSB Living Lab, ett forsknings- och demonstrationshus beläget i Göteborg. I huset bor det runt 40 personer i 29 lägenheter [45]. Utöver att kontinuerligt samla in lastdata utförs flera forskningsprojekt i huset som har över 2000 sensorer för diverse olika projekt [45]. I liknande studier användes även väderdata som temperatur, solinstrålning och nederbörd som indata [15], [16], [28]. Med inspirationen från tidigare studier användes även väderdata som potentiell kandidat till indata i modellerna. Den använda väderdata hämtades ifrån SMHIs väderstation i centrala Göteborg [46].

Tidsupplösningen för väderdatan från SMHI var en timme, vilket resulterade i att datan behövde uppsamplas för att fås på 15-minutersupplösning. Uppsampling av väderdatan gjordes med linjär interpolering.

Då den lastdata som användes i projektet inte var fullständig tillämpades metoder för att komplettera datan och därmed få data av tillräcklig kvalité. Utifrån tillgänglig data valdes perioden 15 mars 2018 - 15 mars 2019. Denna period var det konsekutiva år med minst saknad data.

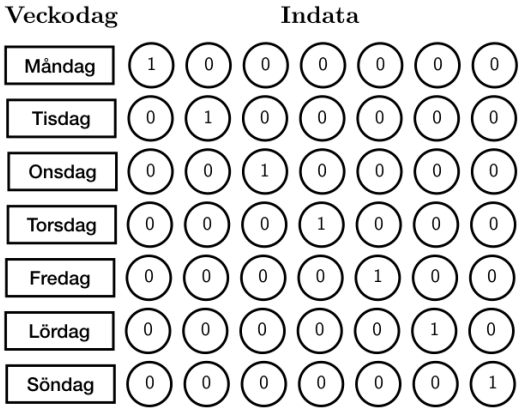
Tidsupplösningen för historisk lastdata var en minut. Eftersom efterfrågad upplösning i modellerna var 15 minuter nedsamplades datan till rätt upplösning. Nedsamplingen utfördes genom att ta medelvärdet för alla minutvärden under aktuell kvart. Som villkor för nedsamplingen bestämdes också att maximalt tre värden fick saknas. Uppfylldes inte detta villkor sattes kvartens värde istället till medelvärdet av föregående och nästkommande kvart.

Om värden för två eller fler på varandra följande kvartar saknades eller om förbrukningsdatan var 0 kW, exkluderades det dygnets data. Utöver dessa, exkluderades också dygnet med tidsomställning till sommartid. Vid tidsomställning till vintertid exkluderades den extra timmen från datan.

Dessa steg genomfördes för att möjliggöra goda prognoser eftersom modellerna kräver sekvensiell och komplett data för att fungera optimalt. Resultatet blev att totalt 975 av 35040 kvartar sattes till medelvärdet av föregående och nästkommande kvart och 23 dagar av 365 exkluderades till följd av saknad data eller tidsomställning.

För att välja vilken indata modellerna skulle ha tillgång till, analyserades olika sorters data i syfte att upptäcka om potentiell data tillförde något. Historisk data för elförbrukning analyserades genom att studera autokorrelation med snabb fouriertransform och partiell autokorrelation med Yule-Walker metoden och utifrån dessa bestämdes vilken historisk data som modellerna ska ha tillgång till. Utöver direkt historisk data analyserades lastdatans periodicitet vidare. Detta gjordes genom att gruppera datan för tid på dygnet och veckodag och visualiera resultatet. För att kunna inkludera denna typ av parametrar som inte nödvändigtvis har linjära samband, implementerades så kallad one-hot-kodning.

One-hot-kodning är en vanlig metod som använts i liknande studier [28], [47], [48] och gör det möjligt för diskreta eller ickelinjära samband att kodas på en form som linjära modeller kan använda. I figur 3.1 presenteras hur one-hot-kodning med hjälp av binära tal kan definiera veckodagar. Principen för one-hot-kodning är enkel att anpassa för att definiera andra kategorier än veckodag, som exempelvis vilken kvart på dygnet det är.



Figur 3.1: Skiss över hur one-hot-kodning översätter veckodag till binära tal vilket senare kan användas som indata för modellerna.

För LSTM-modellen normerades även datan för att dess lager skulle ge optimalt resultat. Detta gjordes med en MinMax-scaler från sklearn-biblioteket. Skalningen normerade datan till intervallet 0-1 enligt

$$x' = \frac{x - \min(X)}{\max(X) - \min(X)}\,, \tag{3.1}$$

där x är en specifik datapunkt och X är all data som normeras och där x' anger det nya, normerade värdet på datapunkten.

Väderdatans potentiella användning i modellerna undersöktes genom att studera Pearsons korrelationkoefficient r, som ges av

$$r = \frac{\mathrm{Cov}(x, y)}{\sigma_x \sigma_y}, \tag{3.2}$$

där x representerar lastdatan, y varierar mellan att representera olika parametrar hos väderdatan som solinstrålning, nederbörd och lufttemperatur, σ är standardavvikelsen och Cov är kovariansen. Genom ekvation (3.2) kunde beslut tas om väderdatan skulle användas i utvecklandet.

3.2 Optimering

Optimeringens huvuduppgift är att anpassa modellerna efter den indata som erhålls för att kunna prestera optimalt. Optimeringsarbete för modellerna skiljer sig något och presenteras vidare nedan.

För den linjära regressionmodellen jämfördes olika typer av linjär regression, där kostnadsfunktionen skiljer dem åt. De metoder som jämfördes är klassisk linjär regression, Ridge-regression, Lasso-regression, PCR och PLSR. Data för träning och testning hölls separat för att kunna utvärdera modellerna korrekt, och samma delning av data gjordes för alla modeller vilket möjliggjorde rättvis jämförelse.

För att undvika överanpassning i ANN- och LSTM-modellerna användes separat validerings-, tränings- och testdata vid träning och utvärdering av modellerna. Efter varje epok utvärderades modellen på valideringsdatan för att se när felet för denna inte längre minskar. En epok är en träningskörning av modellen och avslutas då modellen har sett all träningsdata. Antal epoker begränsades med ett stoppvillkor, sådant att om felet för valideringsdatan inte förbättrades efter ett bestämt antal epoker, avbröts körningen då den antogs ha konvergerat. Detta med förhoppningen att förhindra överträning och minska optimeringstiden. Modellen applicerades därefter på testdatan. Modellen såg således inte testdatan när den tränades.

Vid optimeringen av ANN- och LSTM-modellerna användes så kallad rutnätssökning (eng. grid search). Metoden användes framförallt för att den är lätt att implementera och för att den garanterat ger det bästa valet inom sökområdet [49]. Rutnätssökning är en metod där användaren väljer ett antal parametrar som får olika värden för att därefter pröva alla möjliga kombinationer genom att iterera över dessa och spara resultatet [49]. I slutändan väljs den kombination parametervärden med bäst resultat för att implementeras [49].

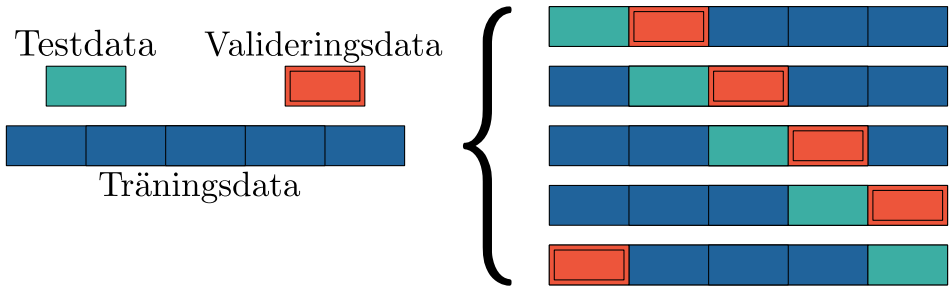
I LSTM och ANN implementerades rutnätssökning med hjälp av nästlade for-loopar där varje for-loop svarade mot en parameter och itererade över valda värden. Parametrarna som valdes för den iterativa processen vid optimering var seriestorlek (eng. batch size), inlärningshastighet, noder och lager. Seriestorlek är antalet datapunkter som ingår i varje serie.

Optimeringen för LSTM-modellen innefattas också av iteration över olika aktiveringsfunktioner. Optimering av ANN avgränsas till aktiveringsfunktionen ReLU (eng. Rectified Linear Unit) då den ses som standardvalet inom de allra flesta tillämpningsområden [50].

För att finna rätt vikter och bias vid träning av ANN- och LSTM-modellerna användes optimeringsfunktionen Adam. Adam är välanvänd inom maskininlärningsoptimering och åstadkommer goda resultat snabbt [51]. Optimeringsfunktionen är en vidareutveckling utav klassisk stokastisk gradientminskning (eng. stochastic gradient decent). Användning av optimeraren Adam medför enkel förändring utav inlärningshastigheten.

3.3 Korsvalidering och prestationsmått

Vid utvärdering av samtliga modeller användes korsvalidering. Korsvalidering är en metod som används för att estimera fel vid regressionsanalys och utvärdera modellen på en större mängd data [52]. Vid korsvalidering delas data upp flera gånger i olika träning och test-uppdelningar [52] som illustrerat i figur 3.2. Korsvalidering av typen utelämna-en (eng. leave one out) användes med samma uppdelning för samtliga modeller, för att få ett jämförbart resultat. Då linjär regression inte använder sig av valideringsdata ingick denna data i träningsdatan för denna modell.



Figur 3.2: Illustration av hur korsvalidering delar upp data, samt utelämna-en-principen.

Korsvalidering ger också fördelen att ett genomsnittligt resultat skapas. Eftersom vikter och noder initieras till slumpvist valda värden, samt att serier kan köras i olika ordning leder det till att ANN- och LSTM-modellerna får något olika resultat på grund av deras stokastiska natur. Genom att modellerna tränas om flera gånger i korsvalideringen fås därför ett medelvärde över dessa modellers resultat som möjliggör en mer rättvis jämförelse.

Som prestationsmått för modellernas resultat användes MAPE (eng. Mean Absolute Percentage Error). MAPE ger ett procentuellt värde på felet för att möjliggöra en relativ jämförelse mellan resultaten. MAPE beräknas enligt

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|, \tag{3.3}$$

där y_i är det faktiska värdet, $\hat{y}_i$ värdet för prognosen och n är antal värden (prognoser). Som kostnadsfunktion för modellerna användes prestationsmåttet MSE (eng. Mean Square Error), dock med en tillagd straffterm för Lasso- och Ridge regression. MSE ger kvadraten av felet och straffar i och med det stora fel mer än om en linjär kostnadsfunktion som MAPE används.

Med samma notation som i ekvation (3.3) definieras MSE enligt

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2. \tag{3.4}$$

där MSE olikt MAPE inte ger ett procentuellt felmått utan speglar kvadraten av det faktiska felmåttet vilket i projektets fall är kilowatt.

3.4 Baslinjemodell och stationär modell

För att utvärdera och ifrågasätta nyttan av de tre AI-modellernas prestation implementerades två primitiva modeller, baslinjemodellen och den stationära modellen. Baslinjemodellen prognostiserar att lasten vid godtycklig tidpunkt är precis lika hög som den var föregående dag vid samma tidpunkt. Den stationära modellen prognostiserar vid godtycklig tidpunkt att lasten för nästkommande fyra kvartar är precis lika hög som senaste kvart. Modellerna utformades på ett sätt som kräver lite beräkningskraft jämfört med AI-modellerna. Felet för baslinjemodellen och den stationära modellen beräknades på samma sätt som för AI-modellerna och mättes i MAPE.

4 Resultat och analys

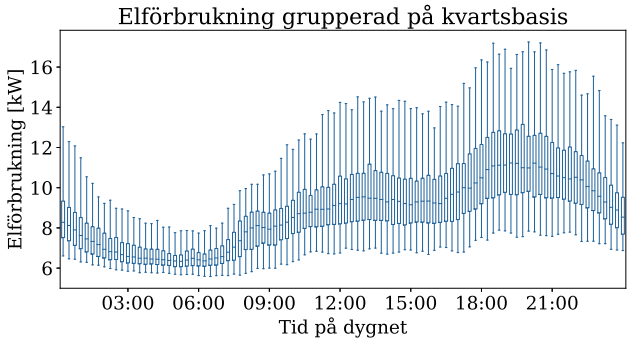
I detta kapitel redovisas och motiveras vilken indata som har använts samt vilken indata som aktivt uteslöts vid utvecklingen av prognostiseringsverktyget. Vidare fastställs resultaten ifrån optimeringsprocessen, som exempelvis seriestorlek och den optimala kombinationen av noder och lager som krävs i syfte att utveckla prognostiseringsverktyget på ett sätt som minimerar felet. Dessutom redogörs för modellernas prestanda genom jämförelser av MAPE för att kunna svara på vilken modell som genererat bäst prognoser. Slutligen studeras även felfördelning av dem tre AI-modellerna för att undersöka om fördelningarna är Gaussiska.

4.1 Indata

I detta delkapitel presenteras analys och bearbetning av indata till modellerna. Detta inkluderar korrelations- och dataanalys av tillgänglig data för att välja inparametrar till modellerna.

4.1.1 Periodiska mönster i datan

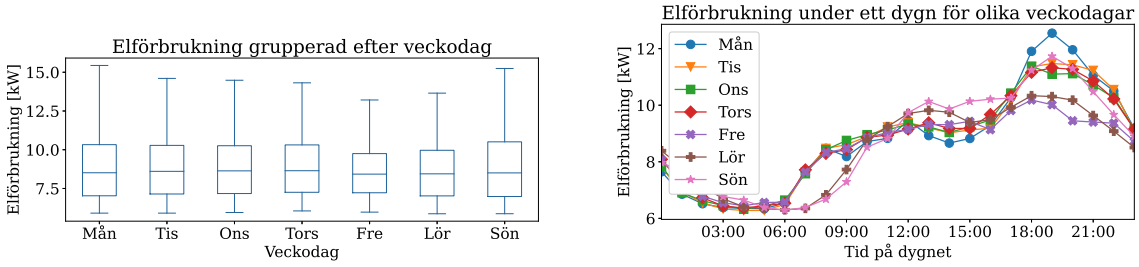
I figur 4.1 visas låddiagram för elförbrukning för perioden 2018-03-15 till 2019-03-15 grupperat på kvartbasis efter tid på dygnet.



Figur 4.1: Låddiagram över elförbrukning grupperat på kvart på dygn. Morrhår tecknar 95% konfidensintervall.

Figur 4.1 indikerar att variationer i elförbrukning knutna till tid på dygnet tycks existera. Från figuren kan det utläsas att medelförbrukning mellan 18:00 och 21:00 är mer än 50 % större än de mellan 03:00 och 06:00. För att fånga detta beteende har one-hot-kodning använts för att indikera tid på dygnet på kvartsbasis för varje given datapunkt. Variationen i elförbrukning skiljer sig också avsevärt beroende på tiden på dygnet men tycks i stora drag följa storleken på lasten, där variationen är större för större last och mindre för mindre last.

I figur 4.2a kan skillnad i den genomsnittliga elförbrukningen grupperat på veckodag utläsas. Även om skillnader i medelförbrukning och variation kan utläsas är skillnaderna mindre än de som kan ses grupperade på tid på dygnet redovisat i figur 4.1. Från figur 4.2b kan dock tydliga skillnader i lastens dygnsrytm utläsas beroende på dag, exempelvis tycks området med lägre last kring småtimmarna vara förskjutet en till två timmar mellan vardag och helg. Vidare tycks även den ökade lasten kring kvällen noterad i figur 4.2b skilja sig med lägre förbrukning på fredagar och lördagar gentemot veckans andra dagar. På grund av detta tillämpas one-hot-kodning även för att indikera veckodag.



(a) Grupperat på veckodag.

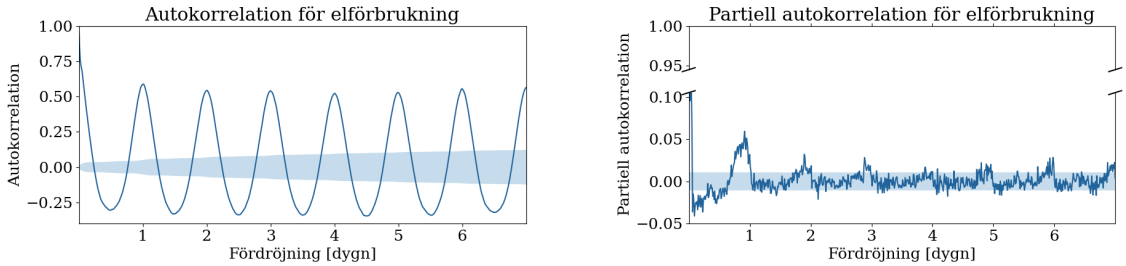
(b) Grupperat på veckodag och timme.

Figur 4.2: Låddiagram över elförbrukning grupperat på veckodag, där morrhår tecknar 95% konfidensinterval. Samt medelvärdet av elförbrukningen grupperat på timme för varje veckodag.

4.1.2 Tidshorisont för historisk data

För att avgöra vilken tidshorisont bakåt som modellerna ska ha tillgång till har autokorrelation och partiell autokorrelation för elförbrukning tagits fram i figur 4.3. Av figur 4.3a ges att lastdatan har en tydligt periodisk autokorrelation, där nuvärdet korrelerar starkt med värden från samma tid på dygnet. Detta förklaras av de tydliga dygnsvariationer noterade i figur 4.1. Av figur 4.3b kan dock utläsas att endast

mycket lite ny information tillförs av data mer än ett dygn gammal. Från detta och det faktum att LSTM-modellen kräver sekvensiell indata bestämdes att lastdata för ett helt dygn tillbaka gavs som indata till modellerna.



(a) Autokorrelation

(b) Partiell Autokorrelation

Figur 4.3: Autokorrelation och partiell autokorrelation för lastdata med upp till sju dygns fördröjning. Autokorrelationen har beräknats med snabb fouriertransform, och den partiella autokorrelationen med Yule-Walker metoden. Ljusare område markerar konfidensintervall om två standardavvikelser.

4.1.3 Väderdata

Då liknande studier har använt väderdata som solinstrålning, nederbörd och lufttemperatur som inparametrar [15], [16], [28], studerades dessa. Pearsons korrelationskoefficient togs fram enligt ekvation (3.2) och kan ses i tabell 4.1 för just de tre parametrarna solinstrålning, nederbörd och lufttemperatur, med data för parametrarna hämtade från SMHIs öppna data för närmast liggande väderstation. På grund av låg korrelation drogs slutsatsen att väderdatan inte skulle bidra till en bättre prognos, och användes därför inte som inparameter till de slutgiltiga modellerna. I slutändan testades modellerna ändå med väderdata som inparametrar men utan att resultatet förbättrades.

Tabell 4.1: Korrelationskoefficienter mellan väderdata ifrån SMHI och lastförbrukning ifrån HSB living lab.

Parameter	Korrelationskoefficient [Pearsons r]
Solinstrålning	-0,0485
Nederbörd	0,01254
Lufttemperatur	0,0204

Den låga korrelationen förklaras dels av att HSB Living Lab under den undersökta perioden inte använde uppvärmnings- och nedkylningsmekanismer som förbrukade stora mängder elektricitet [45]. Vidare antas HSB Living Lab vara för litet som förbrukare för att tydliga mänskliga trender till följd av väder ska existera.

4.2 Optimering

I detta delkapitel kommer resultat av optimering av de olika modellerna att presenteras. Detta innefattar främst vilka parametrar som framtagits som den bästa kombinationen för respektive modell och vilka värden som har testats.

4.2.1 Linjär regression

För att förhindra potentiell överpassning till datan applicerades och testades regulariseringsmetoder utöver vanlig multipel linjär regression. De metoder som testades var Ridge-regression, Lasso-regression, PLSR (eng. Partial Least Square Regression) och PCR (eng. Principal Component Regression). Efter optimering av modellerna observerades dock att ingen av dem presterade bättre än den vanliga linjära regressionsmodellen, endast mycket likt. Då regulariseringsmodellerna innebär en mer påbyggd något mer avancerad modell förväntades det dock att de bör ta något längre tid för både passning och prognostisering. Tränings- och prognostid mättes upp och kan ses i tabell 4.2 där det bekräftas att tiden är längre. På grund av att den vanliga linjära regressionsmodellen både tränades och utförde prognos snabbare än de övriga mer stabila modellerna, samtidigt som den presterade minst lika bra i avseende på fel studerades denna vidare.

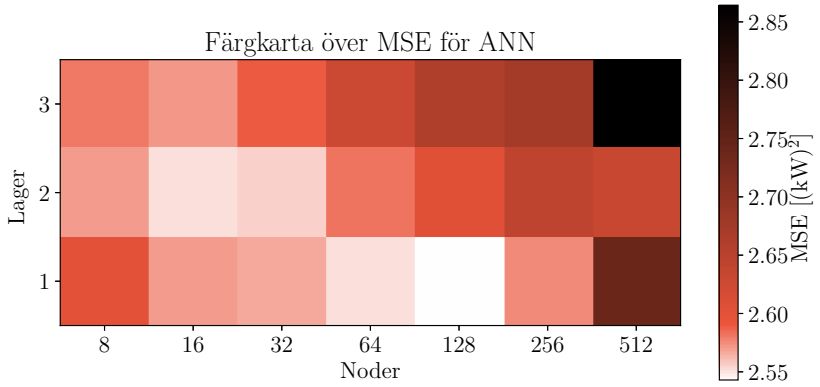
Tabell 4.2: Passnings- och prognostiseringstid för olika linjära regressionsmodeller

Modell	Linjär Regr.	Ridge Regr.	Lasso Regr.	PCR	PLSR
Passningstid [s]	0,380	0,499	2,653	0,916	2,171
Prognostiseringstid [s]	0,078	0,085	0,078	0,128	0,115

Notera att passnings- och prognostiseringstider beror på mängden data som passning respektive prognostisering görs på. Tider i tabell 4.2 avser data för passning och data för prognostisering vid 24624 respektive 8208 tidpunkter.

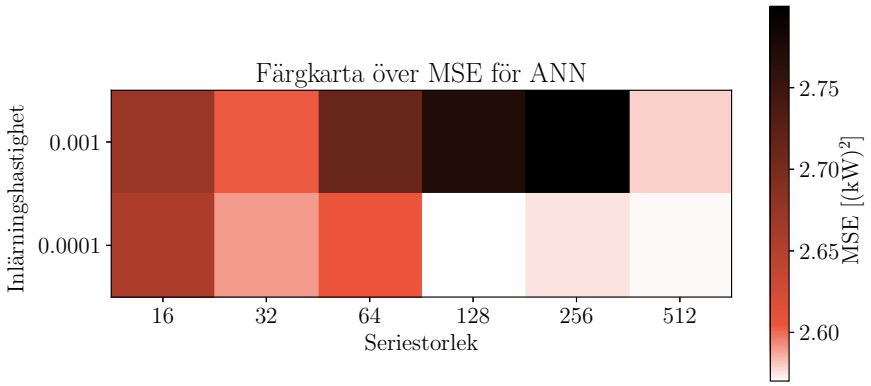
4.2.2 ANN

I ANN-modellen har olika antal lager och olika antal noder undersökts med hjälp av rutnätssökning och användning av valideringsdata. Resultatet av rutnätssökningen hittas i färgkartan i figur 4.4 där antal noder står på horisontalaxeln, antal gömda lager står på vertikalaxeln och färgskalan motsvarar MSE-fel, där lägre MSE-fel motsvarar ljusare färg, och högre MSE-fel motsvarar mörkare färg. Av figurerna kan utläsas att den kombination med lägst fel är kombinationen med ett gömt lager och 128 noder.



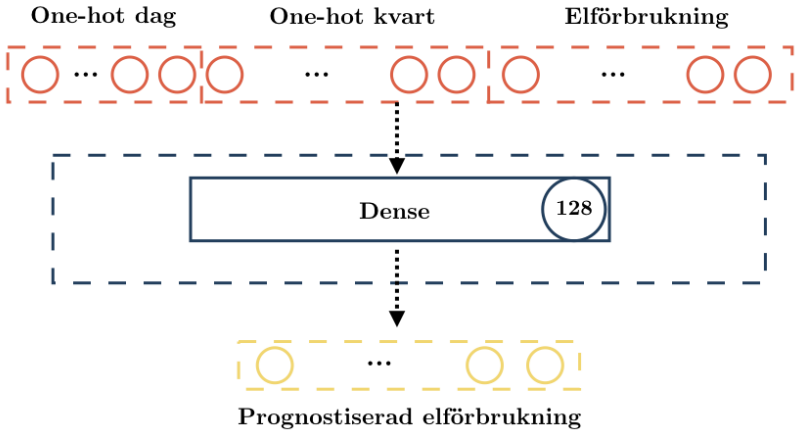
Figur 4.4: Färgkarta för jämförelse av MSE-fel mellan olika kombinationer av parametrarna antal gömda lager och antal noder.

Därefter avgjordes optimal seriestorlek och inlärningshastighet genom en separat rutnätssökning vars resultat presenteras i figur 4.5. Från den kan utläsas att seriestorlek 128 och inlärningshastighet 0.0001 gav bäst resultat.



Figur 4.5: Färgkarta för jämförelse av MSE-fel mellan olika kombinationer av parametrarna seriestorlek och inlärningshastighet.

En enkel figur över den slutgiltiga ANN-modellen ses i figur 4.6.

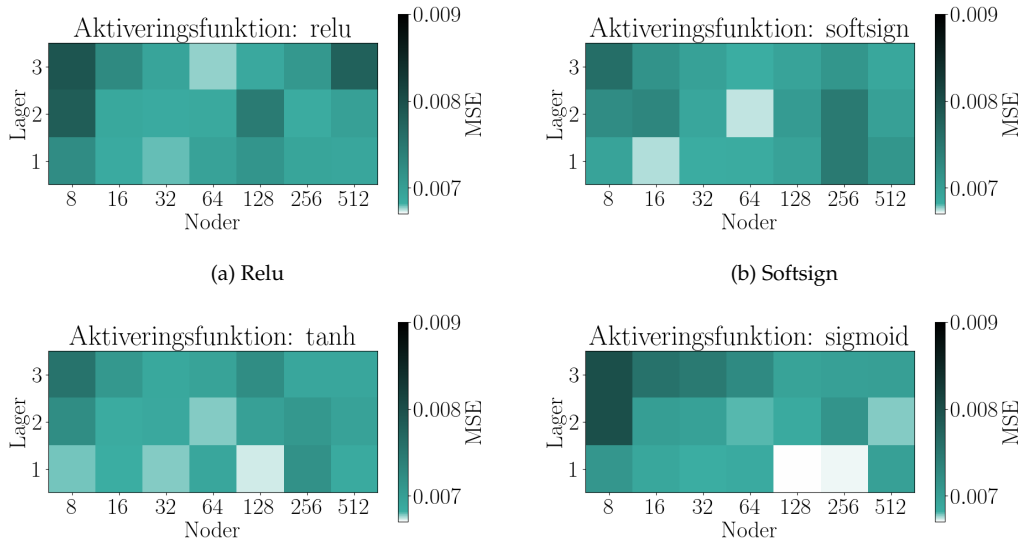


Figur 4.6: Slutkonstruktion för ANN-modellen där cirklar representerar indata och utdata ifrån den färdiga modellen. Det dolda lagret i mitten är uppbyggt av ett Dense-lager på 128 noder.

4.2.3 LSTM

För optimering av LSTM genomfördes en rutnätssökning över olika kombinationer av noder och lager samt ett antal aktiveringsfunktioner enligt figur 4.7. Efter att optimal uppsättning lager, noder och aktiveringsfunktioner bestämts, optimerades seriestorlek och inlärningshastighet, även detta med rutnätssökning.

Resultatet av den första rutnätssökningen, det vill säga den optimala kombinationen av antal LSTM-lager, noder och aktiveringsfunktion, ses i figur 4.7. Figuren visar att ett LSTM lager, 128 noder och sigmoid-aktiveringsfunktion ger lägst MSE.



(a) Relu

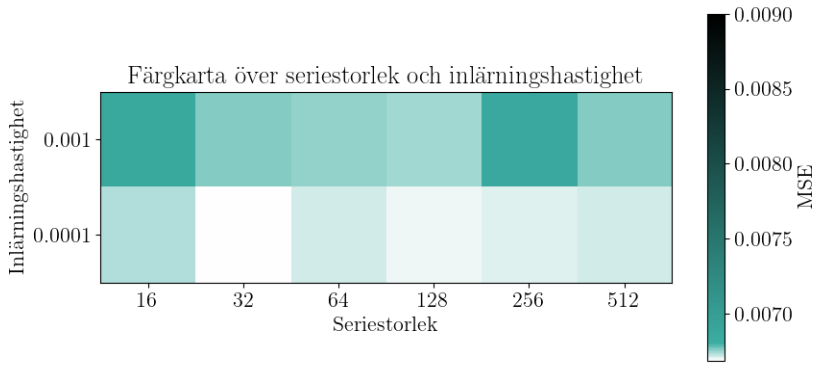
(b) Softsign

(c) Tanh

(d) Sigmoid

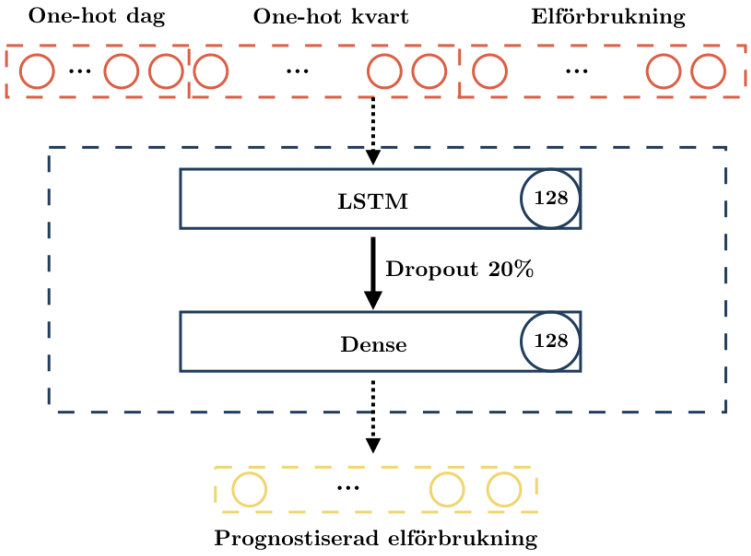
Figur 4.7: Färgkartor för aktiveringsfunktionerna relu, softsign, tanh och sigmoid över fel i MSE vid olika kombinationer av lager och noder på skalad data under optimeringsprocessen. Kombinationen 128 noder, ett LSTM lager och aktiveringsfunktionen sigmoid gav lägst fel. MSE avser här felet på skalad data enligt ekvation (3.1) och saknar enhet.

Därefter optimerades seriestorlek och inlärningshastighet för den modell som gav bäst resultat i den första rutnätssökningen. Resultatet av optimeringen visas i figur 4.8, där det framgår att en kombination av seriestorlek 32 och inlärningshastighet 0,0001 är bra.



Figur 4.8: Fel i MSE med skalad data av körningar med seriestorlekar samt inlärningshastighet. Seriestorlek 32 och inlärningshastighet 0,0001 gav bäst resultat, cirka 0,0067 i MSE. MSE avser här felet på skalad data enligt ekvation (3.1) och saknar enhet.

Slutligen används en Dropout-funktion mellan LSTM-lagren i modellen, vilket gör att ett antal slumpmässigt valda aktiveringar nollställs efter lagret. Detta bidrar till att förhindra överträning då vikterna i nästkommande lager inte blir beroende av ett specifikt invärde. I modellen nollställs 20% av aktiveringarna efter LSTM-lager. Detta gav bättre resultat än utan Dropout. En enkel figur över den slutgiltiga LSTM-modellen ses i figur 4.9.



Figur 4.9: Slutkonstruktion för LSTM-modellen där cirklar representerar indata och utdata ifrån den färdiga modellen. Det dolda lagret i mitten är uppbygt av ett LSTM lager med 128 noder och ett Dense-lager med 128 noder.

4.3 Modellernas prestation

I detta kapitel följer resultat för prognostisering med hjälp av de modeller som konstruerats och studerats i tidigare avsnitt. Resultaten inkluderar prognos för en slumpvist vald dag, genomsnittligt fel för tid på dygn och tid från prognos, statistisk fördelning av fel och en överblick över hur fel beror på tid på dygn.

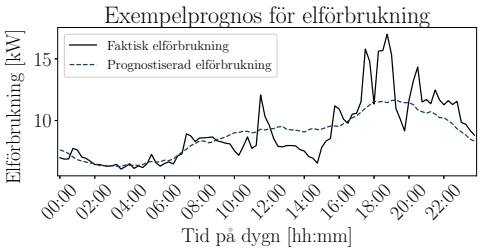
I tabell 4.3 presenteras passnings- och prognostiseringstid för alla tre modeller. Passningstiden avser tiden det tar för den slutgiltiga modellen att passas till sin träningsdata, i det här fallet 3360 datapunkter med 200 värden vardera. Prognostiseringstiden avser tiden för att göra en prognos för 24-timmar framåt, det vill säga 96 prognoser över enskilda kvartar.

Tabell 4.3: Passnings- och prognostiseringstid för modellerna.

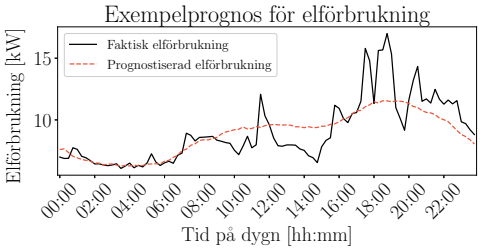
Modell	Passningstid [s]	Prognostiseringstid [s]
Linjär Regression	6,496	0,0009
ANN	17,896	0,058
LSTM	76,996	0,299

Från tabell 4.3 kan observeras att linjär regression har kortast passnings- och prognostiseringstid, därefter ANN och LSTM. Detta speglar också hur komplicerade modellerna är. Tiderna är i detta fall dock såpass korta att det inte har stor påverkan på utfallet av vår jämförelse.

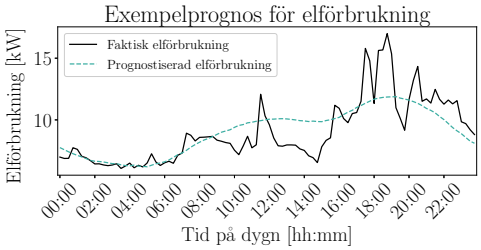
Genom att köra de implementerade modellerna kunde en exempeldag tecknas, där den verkliga lasten skildras i förhållande till den prognostiserade lasten. Exempeldagen valdes ut slumpmässigt och jämförelsen visas i figur 4.10. Där kan observeras att prognoserna trots varierande elförbrukning stundvis passar väl med den faktiska förbrukningen. Dessutom förefaller att modellernas prognoser för nattetid är bättre än prognoserna för senare tider på dagen. Detta antas bero på att förbrukningsvanorna (se figur 4.1) följer en tydligare trend på natten, då konsumenterna ofta sover, medan de under sen eftermiddag fluktuerar kraftigt från dag till dag och har större varians. Det bör dock observeras att figur 4.10 endast avser en utvald dag och att både den faktiska och den prognostiserade förbrukningen i själva verket varierar med val av exempeldagen.



(a) LR



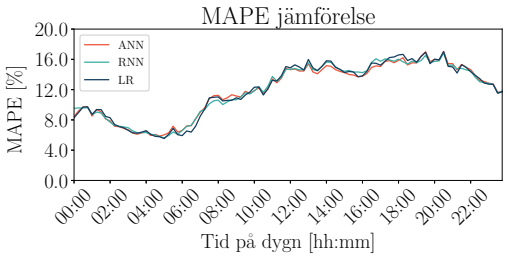
(b) ANN



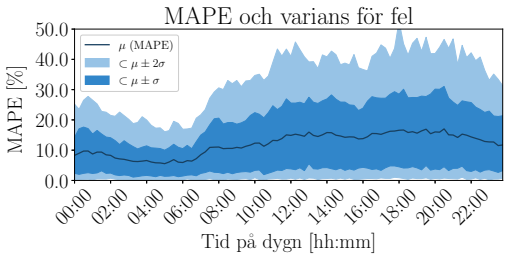
(c) LSTM

Figur 4.10: Exempelprognoser gjorda vid midnatt för alla modeller på en exempeldag.

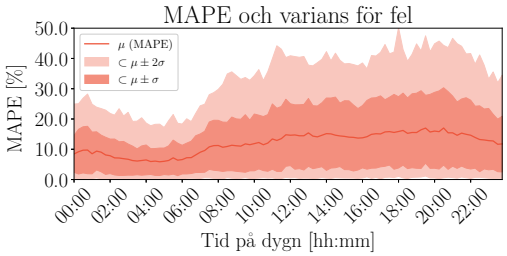
I figur 4.11a jämförs MAPE för samtliga modeller som funktion av tid på dygnet. Dessutom är varje modell tecknad individuellt med det tillhörande konfidensintervallet i figur 4.11b-4.11d. Något anmärkningsvärt är att prognosernas fel befinner sig som lägst runt 03:00-06:00 vilket som tidigare konstaterat tros vara en följd av mindre varians under nattetid. Även variansen av felen är som lägst vid denna tid.



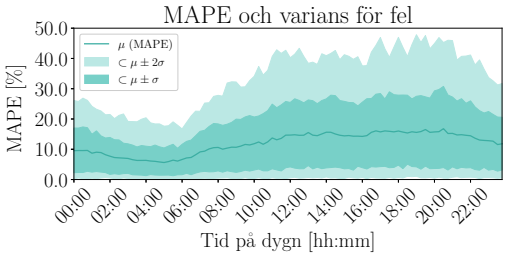
(a) Alla tre modeller



(b) LR med konfidensintervall



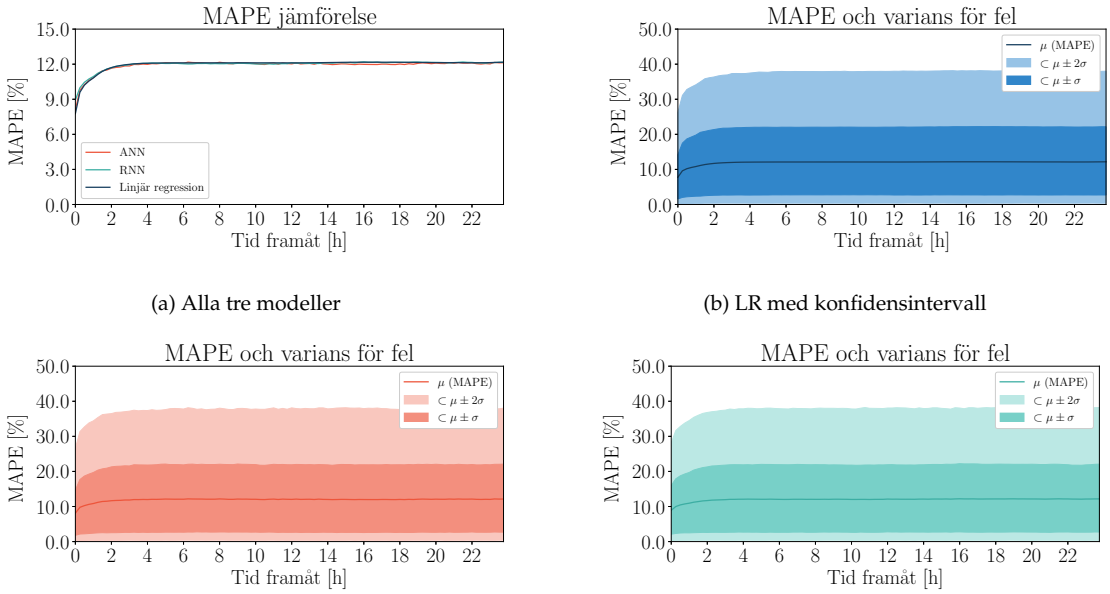
(c) ANN med konfidensintervall



(d) LSTM med konfidensintervall

Figur 4.11: MAPE för given förutspådd kvart ordnat i tid från midnatt. I (a) presenteras en kombination av alla tre modellers MAPE och i (b)-(c) presenteras modellernas MAPE enskilt med konfidensintervall där μ är medelvärdet och σ är en standardavvikelse.

I syfte att åskådliggöra en bild av hur modellerna presterar på prognoser framåt i tiden framställdes i figur 4.12 grafer som beskriver hur MAPE varierar som funktion av den förflutna tiden från starten på prognosen över alla körningar.



(a) Alla tre modeller

(b) LR med konfidensintervall

(c) ANN med konfidensintervall

(d) LSTM med konfidensintervall

Figur 4.12: MAPE för given förutspådd kvart ordnat i tid från då prognos utfördes. I (a) presenteras en kombination av alla tre modellers MAPE och i (b)-(c) presenteras modellernas MAPE enskilt med konfidensintervall där μ är medelvärdet och σ är en standardavvikelse.

En egenskap som samtliga modeller delar är att prognosernas fel till en början är låg men successivt ökar fram tills dess att det stabiliseras runt 11-12% efter ca tre timmar. Det innebär att modellerna prognostiserar förbrukningen i närtid bättre än på längre sikt, vilket tydliggörs i tabell 4.4 där MAPE är tecknad för de första fyra kvartarna för respektive modell.

Att prognostiseringen är bättre de första tre timmarna följer troligen av resultatet ifrån den partiella autokorrelationen i figur 4.3b. Av denna figur ges att de första kvartarna har störst korrelation för prognosen vilket alltså speglas i figur 4.12. Då storleken på felet stabiliseras efter tre timmar, noteras att en prognos utförd närmare i tid troligtvis inte är markant bättre om det värde som är av intresse fortsatt är mer än tre timmar framåt.

Tabell 4.4: MAPE för alla modellers prestanda över de fyra första kvartarna. Sista raden i tabellen presenterar prestanda från en stationär model som behåller utgångsvärdet konstant som prognostisering.

Modell	15 min	30 min	45 min	60 min
LR	7,79 %	9,63 %	10,25 %	10,60 %
ANN	8,10 %	9,77 %	10,25 %	10,06 %
LSTM	8,93 %	9,98 %	10,45 %	10,73%
Stationär	8,28 %	11,14 %	12,43 %	12,26%

I tabell 4.4 presenteras en jämförelse av modellernas MAPE de 4 första kvartarna. Dessutom presenteras en väldigt simpel modell, stationära modellen, för vidare jämförelse. Denna modell tar nuvarande värde för elförbrukning och prognostiserar samma värde för framtida kvartar. ANN och LR modeller presterar bättre än den stationära modellen på den första kvarten, med inte LSTM. För övriga värden presterar alla tre modeller bättre än den stationära modellen.

Från tabell 4.4 kan konstateras att linjär regression ger bäst prognos för 15 minuter framåt. Därefter presterar linjär regression något sämre än ANN- och LSTM-modellerna efter 45 minuter respektive två timmars tid framåt.

I tabell 4.5 är ett genomsnitt utav de tre modellernas MAPE över hela tidshorisonten sammanfattat. Med detta resultat kan en jämförelse mot resultat ifrån tidigare forskning göras.

Tabell 4.5: Genomsnittligt MAPE för alla modeller. Sista raden representerar en Baslinjemodells som tar sin prognos genom att kopiera föregående dag.

Modell	MAPE [%]
LR	12,05 %
ANN	11,91 %
LSTM	11,97 %
Baslinje	17,21 %

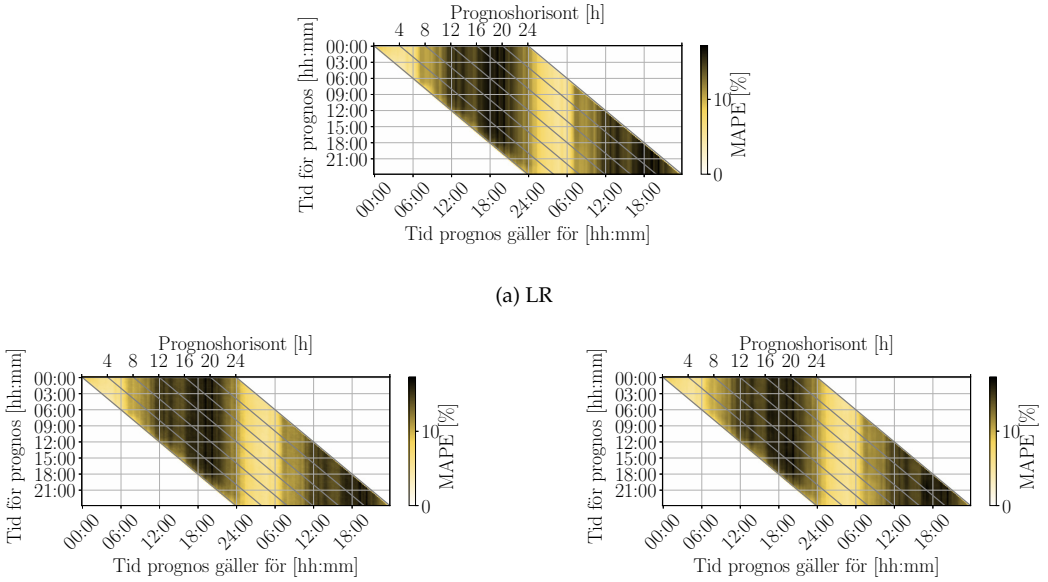
I tabellerna 2.1, 2.2 och 2.3 redovisas MAPE för liknande studier. För linjär regression har fel mellan 1,00-6,10% uppmäts [15], [26]–[28], där [27] som innefattar 784 hushåll och har en upplösning på en timme är den mest relevanta och lika studien av dem. För ANN presenteras MAPE-värden mellan 7,19-20,44% [29]–[31], som innefattar ett byggnadskomplex respektive 69 hushåll, båda med en tidupplösning på en timme och en horisont på 24 timmar och presenterar MAPE-värden på 15,32%

respektive 14,50%. Tidigare LSTM-studier presenterar MAPE-värden mellan 5,35-14,50% [31], [40], [42], [43], där de mest lika studierna är [31], [43] som behandlar 69 hushåll respektive en byggnad, har tidsupplösning på en timme respektive tio minuter, 24 timmars tidshorisont och presenterar MAPE-värden på 8,58% respektive 6,45-14,01%.

Linjär regression presterar alltså sämre än liknande studier, medan ANN- och LSTM-presterar bättre än liknande studier. Det tål att nämnas att den mest lika studien för linjär regression skiljer sig mer från vårt tillämpningsområde än de för ANN och LSTM. Relevansen för modellernas prestanda och resultatet vid jämförelse mot tidigare studier följer i kapitel 5.1.

I tabell 4.5 presenteras även resultatet från baslinjemodellen, som antar samma värde som vid samma tidpunkt föregående dag. Denna resulterar i ett medelfel på 17,21%, vilket alla modeller överträffar med marginal.

I figur 4.13 introduceras ett färgdiagram av hur felet varierar som funktion av både tid på dygnet men även för vilken tid prognosen gjorts. Färgkontrasten på varje punkt i värmekartan motsvarar det procentuella felet i MAPE. Även här kan observeras att felet bland de kortsiktiga prognoserna är lägre än för längre prognoser. Dessutom kan tydliga lodräta mönster urskiljas vilket indikerar att tiden på dygnet har större betydelse för felet än vad tiden mellan prognostillfälle och tiden vars last som prognostiserats har.



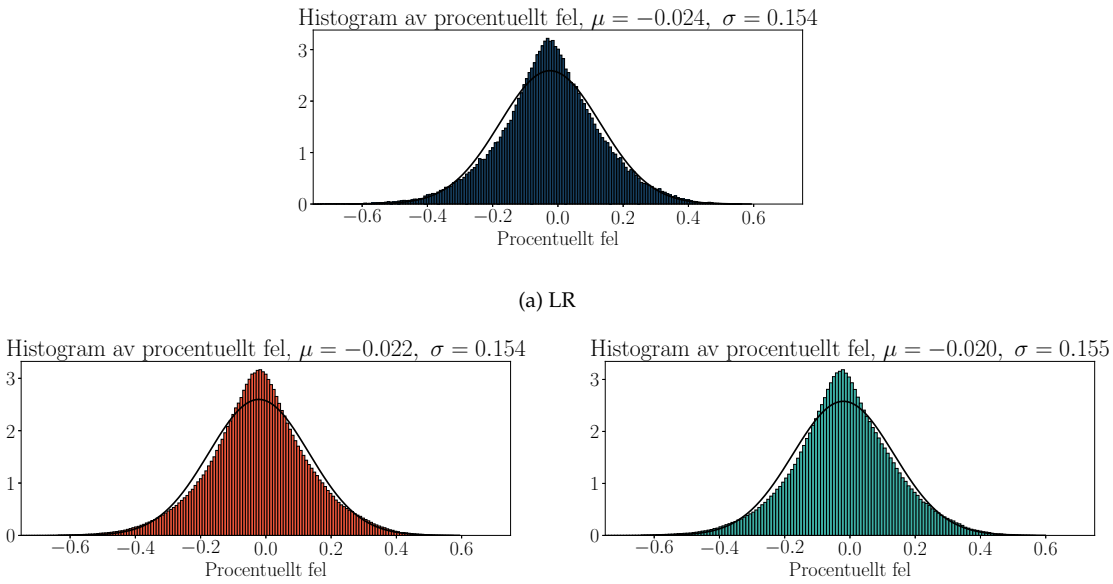
(a) LR

(b) ANN

(c) LSTM

Figur 4.13: Färgdiagram över MAPE med tid för prognostiserat värde och tid för prognos på horisontal- respektive vertikalaxeln. Hur långt fram den förutspådda kvarten är från tiden för prognosen är markerat i ovankant med sneda linjer genom diagramet.

I tidigare studier antas att den bakomliggande sannolikhetsfunktionen som ger upphov till prognosens fel är normalfördelad [15]. För att verifiera detta för projektets modeller har fördelningen för felen uppkomna i tester av modellerna tagits fram och kan ses i figur 4.14. I figuren har även en normalfördelningskurva passad till felen tecknats. Från figuren ses att fördelningen liknar en normalfördelning men inte passar den perfekt.



(a) LR

(b) ANN

(c) LSTM

Figur 4.14: Histogram över sannolikhetstäthet som funktion av procentuellt fel för alla körningar på testdatan, samt heldragen linje för normalfördelning passad till datan. Medelvärdet representeras av μ och σ är en standardavvikelse.

För att få en bättre bild om huruvida dessa skillnader uppkommit av slump eller någon underliggande anledning, togs χ^2 felet mellan de faktiska fördelningarna och normalfördelningen för alla körningar fram och kan ses i tabell 4.6. Från de relativt stora χ^2 felen kan konstateras med stor statistisk säkerhet att felfördelningarna trots att de är mycket lika en normalfördelning inte är uppkommit av perfekt normalfördelad slumpvariabel.

Tabell 4.6: χ^2-fel för passning av felfördelningar för de tre modellerna mot normalfördelning.

Modell	LR	ANN	LSTM
χ^2 fel	8392	8323	8884

5 Diskussion

I detta kapitel diskuteras först resultatet och hur det skiljer sig mot liknande studier samt potentiella anledningar till det. Därefter diskuteras val av metod och dess påverkan på resultatet. Vidare förs även diskussion kring relevans för studien, vad som är nästa steg samt vad modellerna som framtagits kan användas till.

5.1 Studiens bidrag

Vid jämförelse av studiens resultat med tidigare genomförda studier (se tabeller 2.1, 2.2 och 2.3) finnes både skillnader men även likheter. De skillnader som kan observeras tros främst bero på de olika applikationsområdena eller storleken på mätområdet. Vid mätning på större områden kommer de varianser som kan upplevas i ett hushåll att jämna ut varandra, vilket gör det lättare för trender att hittas och för elförbrukning att prognostiseras. Notera att dessa skillnader är stora för linjär regression relativt ANN och LSTM. Detta antas vara upphov till skillnaderna i MAPE mellan denna och tidigare studiers resultat.

Svårighetsgraden av att skapa en prognos påverkas också av upplösning och tidshorisont. Det faktum att arbetet resulterade i prognoser av högre upplösning än tidigare studier typiskt gjort bör alltså leda till större genomsnittligt fel än om samma lägre upplösning utnyttjats. Tidigare forskning [29] pekar på att felet fortsatt bör öka ju längre tidshorisonten för prognosen är. I resultatet framgår dock att vid prognos mer än tre timmar framåt i tiden ökade felet inte väsentligt med ytterligare ökning i tid från då prognos utfördes. Dock är det möjligt att felet hade ökat tydligt om prognosen varit väsentligt längre, men att ökningen är för liten för att märkas över den använda tidshorisonten.

Vidare finns många andra faktorer som kan påverka resultatskillnaderna, exempelvis databehandling innan och efter modellens träning och utvärdering eller modellernas implementation. Det tros dock främst vara de tre tidigarenämnda faktorerna som spelar störst roll för de skillnader som kan observeras, och det är också främst de tre faktorerna som är lättast att jämföra.

På grund av den stationära modellens snabbt växande fel för längre tidshorisonter antas att fyra tidssteg räcker för att visa dess prestation. Det visar sig att både

baslinje- och stationärmodellen presterade sämre än de övriga modeller som studerades. Detta beror sannolikt på deras linjära beteende och avsaknaden av behandling av andra faktorer som lasten korrelerar med.

I slutändan visade det sig att LR-, ANN- och LSTM-modellerna presterade mycket lika. I början av studien förväntades, baserat på resultat från tidigare studier [7], [18], att ANN-modellen skulle prestera bättre än LR-modellen, och att LSTM-modellen skulle prestera bättre än ANN-modellen. Resultatet som finnes av arbetet kan förklaras av att datan inte har några ickelinjära samband, eller att de ickelinjära samband som möjligen finns är såpass otydliga att modellerna inte upptäcker dem. En annan förklaring till att modellerna inte presterade som förväntat skulle kunna vara att datamängden som användes inte är tillräcklig. Tidigare studier som tagits upp i rapporten har dock i många fall inte heller haft mer än ett års historisk data tillgänglig och skillnaden mellan arbetets resultat och dessa rapporter bör alltså inte förklaras av datamängden [27], [29], [30]. Modellerna som utvecklats presterar väsentligt bättre än de två simpla modeller som jämförts med och på en bättre eller likvärdig nivå mot liknande tidigare studier. Till följd av detta goda resultat och att studien behandlar ett relativt outforskat område finnes att projektet producerat ett relevant resultat.

5.2 Metodevaluering

I detta delkapitel diskuteras de metodval som gjorts för projektets utformning. Först diskuteras de metoder som använts för databehandlingen samt avgörande för vilken indata som använts för träningen av modellerna. Vidare följer diskussion kring hur modellerna optimerats för att slutligen analysera de felmått som använts.

5.2.1 Databehandling

Eftersom komplett indata krävs av samtliga modeller, användes metoder som behandlar saknade värden. Detta var nödvändigt för att tillräcklig datamängd till modellernas träning och utvärdering skulle fås. Metoden som användes utformades för att saknad data kunde ersättas med minsta möjliga påverkan på resultatet. Det finns inga vedertagna regler för hur saknad data ska behandlas[53]. Den vanligaste metoden, att helt utesluta saknad data, hade inte fungerat då den skulle ha lett till för

lite kvarstående data. Istället användes medelvärdessubstitution (eng. Mean substitution), dels i samplingen från minut till kvartsupplösning och dels då det saknades en ensam kvart efter sampling. Denna metod riskerar att leda till att variansen i datan underskattas [53]. Dock framgick från analysen av indata att det var mycket hög autokorrelation för data med ett stegs fördröjning och därmed borde medelvärdessubstitition fungera väl. I de fall där det saknades mer än en kvarts data i rad uteslöts istället hela dagen. Här uppstår en balansgång där ett högt ställt krav leder till låg mängd data men av hög kvalité och ett lågt ställt krav leder till högre mängd data av sämre kvalité.

Regler för gränser och villkor vid förbehandling av datan valdes något arbiträrt genom att observera hur slutgiltig data skulle påverkas av olika värden. Bedömningen som gjordes är att de regler som användes har gjort det möjligt för datan från HSB living lab att användas utan någon nämnvärd påverkan på resultatet. I en ideal situation hade naturligtvis ett helt komplett dataset varit att föredra men i verkligheten krävs ofta dessa typer av metoder [53].

En viktig aspekt att belysa inom valet av metod är kopplat till hur det slutgiltiga resultatet påverkades av korsvalideringen. Som diskuterat i kapitel 3.3 användes korsvalideringen för att kunna ge ett mer rättvist resultat av de olika modellernas träning eftersom deras prestanda varierar beroende på vilken indata som använts. Korsvalidering som den utförts i arbetet speglar dock en del overkliga fall. Med overkliga menas att modellerna i vissa scenarion av korsvalideringen evaluerades och testades på data som är daterat framåt i tiden relativt träningsdatan och valideringsdatan. Styrkan med korsvalidering är att den ger en bild över hur modellerna presterar över all data. Detta då val av endast en testdatauppdelning kan ge missvisande prestanda över hur modellen fungerar eftersom att just den valda testdatan kan vara enkel eller svår att prognostisera. Att korsvalidering utförts motiveras genom att den tidigare effekten av att modellen tekniskt sett kunde se i framtiden inte antas påverka lika mycket som den senare.

5.2.2 Optimering

Vad gäller val av optimeringsparametrar, kanske främst inom ANN- och LSTM-optimering är det svårt att säga om en bra uppsättning värden valdes att optimera över. Detta då det för maskininlärning inte finns något vedertaget sätt eller perfekt

metod för att få fram parametervärden, utan svaret på frågan om vilka värden som är bra blir ofta att testa sig fram. Detta beror delvis på att det är väldigt olika från fall till fall och kan bero starkt på indata och applikationsområde.

Något annat som tål att diskuteras är användningen av rutnätssökning som optimeringsalgoritm. Då principen för den innebär att modellen tränas för varje kombination av olika parametrar som önskas undersökas tar rutnätssökning lång tid om många olika parameterkombinationer önskas testas. Eftersom inte alla möjliga kombinationer av variabler kan testas då detta skulle ta för lång tid riskerar den mest optimala kombinationen att missas. Vidare är den rutnätssökning som implementerades icke-dynamisk på grund av den tid som algoritmen kräver. Detta innebär att antalet noder inte har varierats mellan de gömda lagren. Vid användning av en annan optimeringsalgoritm som inte testar alla möjliga varianter utan istället gör kvalificerade gissningar om vilken kombination som ska testas härnäst hade möjligen en bättre kombination kunnat finnas med samma datorkraft.

5.2.3 Prestationsmått

I denna rapport har framförallt MAPE-värden presenterats för jämförelse av olika modeller, men även för utvärdering av de modeller som tagits fram. MAPE är ett mått som kommer med både nackdelar och fördelar. Nackdelar med MAPE innefattar att det är ett procentuellt mått, vilket innebär att oändligt små värden inte kan fås på samma sätt som det kan få oändligt stora. Detta problem har lösts genom träning med MSE som är ett annat felmått. Notera att då modellerna tränas med ett annat mått än vad som presenteras, kommer troligen inte det optimala resultatet med avseende på presentationsmåttet att presenteras.

Däremot finns det fördelar med MAPE-måttet, och de innefattas bland annat av att det är ett väldigt bra jämförelsemått för jämförelse med andra studier. Det är svårt att hitta studier som utfördes på exakt samma sorts data eller skala som i detta arbete, vilket gör det svårt för värden som till exempel MSE att jämföras, då det är svårt att veta om ett visst MSE-värde är bra och ett annat inte. Vid jämförelse av exempelvis lastprognostisering av en stad där elektriska laster hanteras på terawatt-skalan, och ett bostadsområde där laster hanteras på megawatt-skalan är det väldigt svårt att jämföra exempelvis MSE-mått, eftersom de är väldigt olika i storlek. Med MAPE-värden underlättas jämförelsen istället markant.

5.3 Vidareutveckling

För att bygga vidare på arbetet och förbättra produkten och de resultat som nåtts, presenteras i följande delkapitel potentiella vidareutvecklingar. Dessa inkluderar vidareutveckling av koden, framställande av användargränssnitt och slutligen integrering med eltillverkning och elmarknad för att göra beslut om energihantering inom elsystemet.

Vidareutveckling av produkten kan utföras på flera områden. Ett område som diskuterats tidigare är metoden för optimering. I nuläget används rutnätssökning. En ökning i antal undersökta värden och kombinationer möjliggör eventuella förbättringar i prestanda. Utöver en snabbare optimeringsmetod kan även dynamiken av den förbättras, exempelvis genom att vid användning av flera lager kunna undersöka olika många noder och/eller olika aktiveringsfunktioner i de olika lagren.

I denna studie har endast tre AI-modeller optimerats och analyserats. Ytterligare ett område för vidareutveckling är eventuell undersökning av fler modeller alternativt fler varianter av de presenterade modellerna. Detta utökar chanserna att finna en modell där vissa samband kan uppfattas som inte uppfattats av de studerade modellerna och därmed få ett bättre resultat. I några tidigare studier har speciella varianter av de modeller som analyserats i denna studie presenterats och i vissa fall har dessa presterat bättre [43].

Vidare kan produkten också vidareutvecklas genom bättre analys av data och fler potentiella indata, för att försöka upptäcka samband som i denna studie inte upptäckts. Vid nyupptäckta samband kan modellerna matas med dessa indataparametrar och potentiellt prestera bättre.

En ytterligare utvecklingsmöjlighet föreligger i att användarvänligheten i produkten förbättras så att det kan brukas av fler användare i syfte att större samhällsnytta på konsumentnivå kan göras. Bland annat kan produkten generaliseras till någon form av applikation med tillhörande användargränssnitt, där inga förkunskaper eller insikter i utvecklingsprocessen krävs. Dessutom kan en generalisering göras som tillåter programmet att hantera dataset med annan tidsupplösning än den 15-minutersupplösning som varit aktuell under projektet. Även tidshorisonten skulle kunna anpassas efter användarens begäran.

För att kunna ta kvalificerade beslut om energihantering som att starta eller avvakta med energikrävande processer och lagra eller utnyttja energin som produceras vid en given tidpunkt krävs både last- och produktionsprognoser. En vidareutveckling av projektet skulle således kunna var att ett prognosverktyg för produktion från intermittenta energikällor utvecklas. Vidare kan det med dessa verktyg implementeras en automatiserad beslutsprocess för energihantering av småskaliga energisystem där, baserat på last- och produktionsprognoser och eventuellt aktuella elpriser, beslut om lagring eller försäljning av egengenererad el tas i realtid av programmet.

6 Slutsatser

Genom att utnyttja färdiga programbibliotek utvecklade för databashantering och uppbyggnad av maskininlärningsmodellerna har tre AI-baserade prognostiseringsprogram utvecklats. Dessa inkluderar linjär regression, ANN och RNN med LSTM. För optimering och träning av modellerna användes historisk lastdata från HSB Living Lab. Modellernas prestation validerades också på samma datamängd.

De tre metoderna som implementerades presterade mycket lika i utvärderingssteget. Skillnaden i genomsnittlig MAPE för alla modellerna på all utvärderingsdata var endast 0,19 procentenheter. Vad gäller fördelningen hos felen för de tre modellerna tedde sig denna i alla tre fall vara lik en Gaussisk fördelning. Med stor statistisk säkerhet kan dock konstateras att fördelningarna inte uppkommit av en helt Gaussisk slumpvariabel. Likväl kan fördelningarna troligtvis i de flesta praktiska applikationer approximeras väl med en Gaussisk fördelning.

Det artificiella neuronnätet (ANN) hade dock något mindre fel än de två andra modellerna. I tidigare jämförande forskning har LSTM ofta presterat väsentligt bättre än de två andra modellerna på liknande prognostiseringsproblem fast av större skala. Dock poängteras i tidigare forskning att för att respektive modell ska kunna fånga beteenden i datan krävs data av god kvalité, med tydliga samband samt låg slumpmässighet. Det faktum att de mer avancerade ANN- och LSTM-modellerna inte presterar långt mycket bättre än linjär regression förklaras ergo av att det dataset utvärderingen är gjord på inte har tillräckligt tydliga samband, alternativt för hög slumpmässighet kring dessa. Vid lastprognostisering av småskaliga energisystem kan alltså mindre avancerade modeller som linjär regression prestera jämförbart med mer avancerade som ANN och RNN med LSTM. Detta då datan löper ökad risk för att vara för brusig för att de senare ska kunna utnyttja mer komplicerade samband mellan in- och utdata.

Referenser

[1] United Nations, ”Climate Change,” jan. 2016, [Online]. Tillgänglig: https://www.un.org/en/climatechange/paris-agreementl (hämtad 2021-02-04).

[2] Naturvårdsverket, *Parisavtalet*, [Online]. Tillgänglig: https://www.naturvardsverket.se/parisavtalet (hämtad 2021-02-04).

[3] European Environment Agency, ”Energy and climate change,” 2017, [Online]. Tillgänglig: https://www.eea.europa.eu/signals/signals-2017/articles/energy-and-climate-change (hämtad 2021-02-04).

[4] A. Vourvoulias, ”Pros and Cons of Solar Energy,” *Green Match*, mars 2021, [Online]. Tillgänglig: https://www.greenmatch.co.uk/blog/2014/08/5-advantages-and-5-disadvantages-of-solar-energy (hämtad 2021-02-04).

[5] G. Ren, J. Liu, J. Wan, Y. Guo och D. Yu, ”Overview of wind power intermittency: Impacts, measurements, and mitigation solutions,” *Applied Energy*, vol. 204, s. 48, 2017.

[6] ”Google Search trends,” 2020, [Online]. Tillgänglig: https://trends.google.com/trends/explore?date=today%5C%205-y&q=%5C%2Fm%5C%2F0mkz,%5C%2Fm%5C%5C%2F01hyh_ (hämtad 2021-02-04).

[7] N. Razavian, F. Knoll och K. J. Geras, ”Artificial Intelligence Explained for Nonexperts,” *Seminars in Musculoskeletal Radiology*, vol. 24, nr 1, ss. 3–11, febr. 2020. DOI: 10.1055/s-0039-3401041.

[8] K. Kumar och G. S. M. Thakur, ”Advanced Applications of Neural Networks and Artificial Intelligence: A Review,” *International Journal of Information Technology and Computer Science*, vol. 4, nr 6, ss. 57–68, juni 2012. DOI: 10.5815/ijitcs.2012.06.08.

[9] A. Amini, ”MIT Introduction to Deep Learning | 6.S191,” *YouTube*, febr. 2020, [Video]. Tillgänglig: https://www.youtube.com/watch?v=njKP3FqW3Sk.

[10] S. Das, A. Dey, A. Pal och N. Roy, ”Applications of Artificial Intelligence in Machine Learning: Review and Prospect,” *International Journal of Computer Applications*, vol. 115, nr 9, ss. 31–41, april 2015. DOI: 10.5120/20182-2402.

[11] A. Almalaq och G. Edwards, ”A Review of Deep Learning Methods Applied on Load Forecasting,” i *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Cancún, Mexico, 2017, ss. 511–516. DOI: 10.1109/ICMLA.2017.0-110.

[12] L. Nespoli, V. Medici, K. Lopatichki och F. Sossan, ”Hierarchical demand forecasting benchmark for the distribution grid,” *Electric Power Systems Research*, vol. 189, s. 106 755, 2020.

[13] H. Shi, M. Xu och R. Li, ”Deep learning for household load forecasting—A novel pooling deep RNN,” *IEEE Transactions on Smart Grid*, vol. 9, nr 5, ss. 5271–5280, 2017.

[14] A. Ghanbari, E. Hadavandi och S. Abbasian-Naghneh, ”Comparison of artificial intelligence based techniques for short term load forecasting,” i *2010 Third International Conference on Business Intelligence and Financial Engineering*, Hong Kong, Kina, 2010, ss. 6–10.

[15] N. Amral, C. Özveren och D. King, ”Short term load forecasting using multiple linear regression,” i *2007 42nd International universities power engineering conference*, IEEE, Brighton, England, 2007, ss. 1192–1198.

[16] S. Singh, S. Hussain och M. A. Bazaz, ”Short term load forecasting using artificial neural network,” i *2017 Fourth International Conference on Image Information Processing (ICIIP)*, Shimla, Indien, 2017, ss. 1–5. DOI: 10.1109/ICIIP.2017.8313703.

[17] S. M. Elgarhy, M. M. Othman, A. Taha och H. M. Hasanien, ”Short term load forecasting using ANN technique,” i *2017 Nineteenth International Middle East Power Systems Conference (MEPCON)*, Kairo, Egypten, dec. 2017, ss. 1385–1394. DOI: 10.1109/MEPCON.2017.8301364.

[18] A. Ghanbari, E. Hadavandi och S. Abbasian-Naghneh, ”Comparison of Artificial Intelligence Based Techniques for Short Term Load Forecasting,” i *2010 Third International Conference on Business Intelligence and Financial Engineering*, 2010, ss. 6–10. DOI: 10.1109/BIFE.2010.12.

[19] M. Q. Raza och A. Khosravi, ”A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings,” *Renewable and Sustainable Energy Reviews*, vol. 50, ss. 1352–1372, juni 2015. Tillgänglig: https://www.researchgate.net/publication/281786549_A_review_on_artificial_intelligence_based_load_demand_forecasting_techniques_for_smart_grid_and_buildings (hämtad 2021-02-08).

[20] B. Biering, ”GETTING STARTED WITH AI: HOW MUCH DATA DO YOU NEED?” *2021.ai*, jan. 2020, [Online]. Tillgänglig: https://2021.ai/getting-started-ai-how-much-data-needed/ (hämtad 2021-02-05).

[21] D. Soni, ”Supervised vs. Unsupervised Learning,” *towards data science*, Tillgänglig: https://towardsdatascience.com/supervised-vs-unsupervised-learning-14f68e32ea8d (hämtad 2021-02-28).

[22] D. C. Montgomery, E. A. Peck och G. G. Vining, *Introduction to linear regression analysis*, 6. uppl. Hoboken, NJ, USA: John Wiley & Sons, 2021.

[23] I. Goodfellow, Y. Bengio och A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. Tillgänglig: http://www.deeplearningbook.org.

[24] G. A. Seber och A. J. Lee, *Linear regression analysis*, 2. uppl. Hoboken, NJ, USA: John Wiley & Sons, 2012.

[25] S. Wold, M. Sjöström och L. Eriksson, ”PLS-regression: a basic tool of chemometrics,” *Chemometrics and intelligent laboratory systems*, vol. 58, nr 2, ss. 109–130, 2001.

[26] G. Dudek, ”Pattern-based local linear regression models for short-term load forecasting,” *Electric power systems research*, vol. 130, ss. 139–147, jan. 2016. DOI: 10.1016/j.epsr.2015.09.001.

[27] S. Humeau, T. K. Wijaya, M. Vasirani och K. Aberer, ”Electricity load forecasting for residential customers: Exploiting aggregation and correlation between households,” i *2013 Sustainable internet and ICT for sustainability (SustainIT)*, IEEE, 2013, ss. 1–6.

[28] T. Hong, P. Wang och H. L. Willis, ”A naïve multiple linear regression benchmark for short term load forecasting,” i *2011 IEEE Power and Energy Society General Meeting*, IEEE, Detroit, Michigan, USA, 2011, ss. 1–6.

[29] Y. Liu, W. Wang och N. Ghadimi, ”Electricity load forecasting by an improved forecast engine for building level consumers,” *Energy*, vol. 139, ss. 18–30, 2017. DOI: 10.1016/j.energy.2017.07.150.

[30] S. J. Baek och S. G. Yoon, ”Short-Term Load Forecasting for Campus Building with Small-Scale Loads by Types Using Artificial Neural Network,” i *2019 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, Washington, DC, USA, 2019, ss. 1–5. DOI: 10.1109/ISGT.2019.8791674.

[31] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu och Y. Zhang, ”Short-Term Residential Load Forecasting Based on LSTM Recurrent Neural Network,” *IEEE Transactions on Smart Grid*, vol. 10, nr 1, ss. 841–851, jan. 2019. DOI: 10.1109/TSG.2017.2753802.

[32] A. K. Jain, J. Mao och K. M. Mohiuddin, ”Artificial neural networks: a tutorial,” *Computer*, vol. 29, nr 3, ss. 31–44, 1996. DOI: 10.1109/2.485891.

[33] D. P. Kingma och J. Ba, ”Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[34] K. Park, J. Kim och J. Lee, ”Visual Field Prediction using Recurrent Neural Network,” *Scientific reports*, vol. 9, nr 1, ss. 1–12, 2019. DOI: 10.1038/s41598-019-44852-6.

[35] Y. Bengio, P. Simard och P. Frasconi, ”Learning long-term dependencies with gradient descent is difficult,” *IEEE Transactions on Neural Networks*, vol. 5, nr 2, ss. 157–166, mars 1994. DOI: 10.1109/72.279181.

[36] G. V. Houdt, C. Mosquera och G. Nápoles, ”A review on the long short-term memory model,” *Artificial Intelligence Review*, vol. 53, 2020. DOI: https://doi.org/10.1007/s10462-020-09838-1.

[37] Y. Yu, X. Si, C. Hu och J. Zhang, ”A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures,” *Neural Computation*, vol. 31, nr 7, ss. 1235–1270, 2019. DOI: 10.1162/neco_a_01199.

[38] S. Motepe, A. N. Hasan och R. Stopforth, ”Improving Load Forecasting Process for a Power Distribution Network Using Hybrid AI and Deep Learning Algorithms,” *IEEE Access*, vol. 7, ss. 82 584–82 598, juni 2019. DOI: 10.1109/ACCESS.2019.2923796.

[39] S. Muzaffar och A. Afshari, ”Short-Term Load Forecasts using LSTM Networks,” *Energy Procedia*, vol. 158, ss. 2922–2927, 2019. DOI: https://doi.org/10.1016/j.egypro.2019.01.952.

[40] R. K. Agrawal, F. Muchahary och M. M. Tripathi, ”Long term load forecasting with hourly predictions based on long-short-term-memory networks,” i *2018 IEEE Texas Power and Energy Conference (TPEC)*, College Station, TX, USA, 2018, ss. 1–6.

[41] C. Liu, Z. Jin, J. Gu och C. Qiu, ”Short-term load forecasting using a long short-term memory network,” i *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, Turin, Italy, 2017, ss. 1–6.

[42] J. Zheng, C. Xu, Z. Zhang och X. Li, "Electric Load Forecasting in Smart Grid Using Long-Short-Term-Memory based Recurrent Neural Network," i *2017 51st Annual Conference on Information Sciences and Systems (CISS)*, Baltimore, MD, USA, 2017, ss. 1–6.

[43] Y. Xu *et al.*, "Load Forecasting Method for Building Energy Systems Based On Modified Two-Layer LSTM," i *2021 3rd Asia Energy and Electrical Engineering Symposium (AEEES)*, Chengdu, China, 2021, ss. 660–665.

[44] A. C. Müller och S. Guido, *Introduction to machine learning with Python: A guide for data scientists*, 1. uppl. Sebastopol, CA, USA: O'Reilly Media, Inc., 2016.

[45] HSB, "HSB Living Lab", 2021. Tillgänglig: `https://www.hsb.se/hsblivinglab` (hämtad 2021-02-04).

[46] Sveriges meteorologiska och hydrologiska institut (SMHI), "Data", 2021, [Online]. Tillgänglig: `https://www.smhi.se/data` (hämtad 2021-04-30).

[47] Z. Yu, Z. Niu, W. Tang och Q. Wu, "Deep learning for daily peak load forecasting–a novel gated recurrent neural network combining dynamic time warping," *Ieee Access*, vol. 7, ss. 17 184–17 194, 2019.

[48] D. Gan, Y. Wang, S. Yang och C. Kang, "Embedding based quantile regression neural network for probabilistic load forecasting," *Journal of Modern Power Systems and Clean Energy*, vol. 6, nr 2, ss. 244–254, 2018. DOI: `10.1007/s40565-018-0380-x`.

[49] J. Bergstra och Y. Bengio, "Random Search for Hyper-Parameter Optimization," *Journal of machine learning research*, vol. 13, nr 2, ss. 281–305, febr. 2012.

[50] P. Ramachandran, B. Zoph och Q. V. Le, "Searching for activation functions," *arXiv preprint arXiv:1710.05941*, 2017. Tillgänglig: `https://arxiv.org/abs/1710.05941`.

[51] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016. Tillgänglig: `http://arxiv.org/abs/1609.04747`.

[52] S. Arlot och A. Celisse, "A survey of cross-validation procedures for model selection," *Statistics Surveys*, vol. 4, ss. 40–79, 2010. DOI: `10.1214/09-SS054`.

[53] D. A. Bennett, "How can I deal with missing data in my study?" *Australian and New Zealand Journal of Public Health*, vol. 25, nr 5, ss. 464–469, okt. 2001. DOI: `10.1111/j.1467-842X.2001.tb00294.x`.