Learning production functions for supply chains with graph neural networks

Departamento de Ciencia de la Computación

IIC3696 - Tópicos Avanzados en Aprendizaje de Máquina

Presentador: Gabriel Catalán

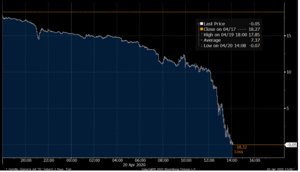
10/09/2024

¿Por qué elegí este paper?

Nassim Nicholas Taleb @nntaleb · Apr 20, 2020

Every #RWRI we discuss the situation how *oil above ground* (hence the front oil future) can sqeeze suckers and have a negative value because it can be impossible to store during a **glut** and impossible to abandon because of EPA.
Negative optionality from liability.

> **Bloomberg** @business · Apr 20, 2020
>
> BREAKING: WTI crude oil futures trade at negative price for first time trib.al/wzlc6mY



THE PANDEMIC ISN'T A BLACK SWAN BUT A PORTENT OF A MORE FRAGILE GLOBAL SYSTEM

By Bernard Avishai

April 21, 2020

Nassim Nicholas Taleb says that his profession is "probability." But his vocation is showing how the unpredictable is increasingly probable. Photograph Michael Appleton / NYT / Redux

Nassim Nicholas Taleb @nntaleb

The Suez Canal, a generalization of:
"The market is a large movie theater with a small door" (Dynamic Hedging).

#Nonlinarities #Fragility x.com/nameshiv/statu...

This post is unavailable.

12:10 PM · Mar 24, 2021

Nassim Nicholas Taleb @nntaleb · Sep 11, 2021

I've seen gluts not followed by shortages, but I've never seen a shortage not followed by a **glut**.

Autores

Serina Chang[1], Zhiyin Lin[1], Benjamin Yan[1], Swapnil Bembde[2], Qi Xiu[2], Chi Heem Wong[1,2], Yu Qin[2,3], Frank Kloster[2], Alex Luo[2], Raj Palleti[1,2], and Jure Leskovec[1]

[1]Stanford University, Department of Computer Science
[2]Hitachi America, Ltd.
[3]Tulane University

Resumen

- Economía global
- Funciones de producción
- Nuevo modelo
- Simulador
- Resultados

Introducción

- Importancia
- *Temporal production graphs* - TPGs
- Nueva clase de GNNs

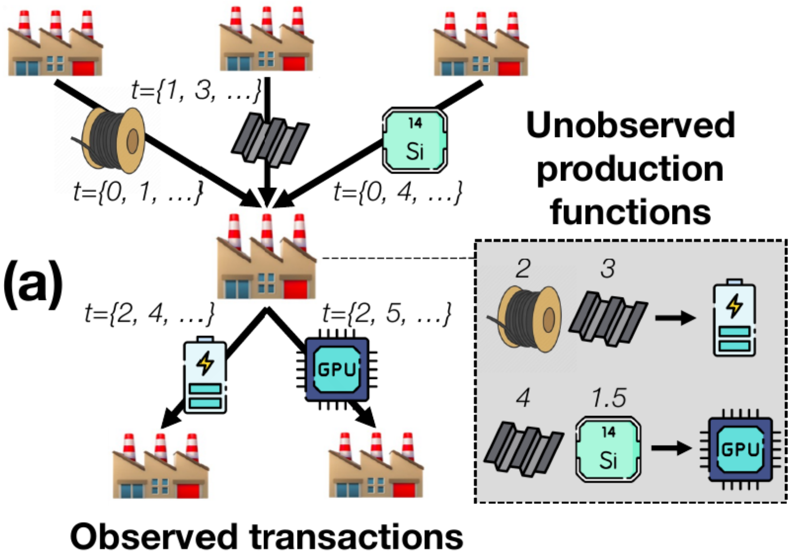
Introducción

1. **Problema**
2. **Modelos**
3. **Datos**
4. **Resultados**

Aprendiendo de TPGs

1. **Definición del problema**
2. **Arquitectura del modelo**
3. **Entrenamiento y evaluación**

Definición del problema



Definición del problema

- Set de transacciones $\mathcal{T}$
- Grafo $G_{\text{txns}} = \{\mathcal{N}, \mathcal{E}\}$
- Nodos $\mathcal{N}$, n firmas y m productos
- Aristas $\mathcal{E} := \{e(s, b, p, t)\}$
- Función $\mathcal{F}_p : \mathbb{R}_+ \to \mathbb{R}_+^m$
- Grafo G_{prod}

Arquitectura del modelo

1. **Módulo inventario**
2. **SC-TGN**
3. **SC-GraphMixer**
4. **Decoder**

Módulo inventario

$$\text{buy}(i, p, t) = \sum_{e(s,i,p,t) \in \mathcal{E}} \text{amt}(s, i, p, t) \quad (1)$$

$$\text{consume}(i, p, t) = \sum_{e(i,b,p_s,t) \in \mathcal{E}} \alpha_{p_s p} \cdot \text{amt}\left(i, b, p_s, t\right) \quad (2)$$

$$\mathbf{x}_i^{(t+1)} = \max\left(0, \mathbf{x}_i^{(t)} + \mathbf{b}_i^{(t)} - \mathbf{c}_i^{(t)}\right) \quad (3)$$

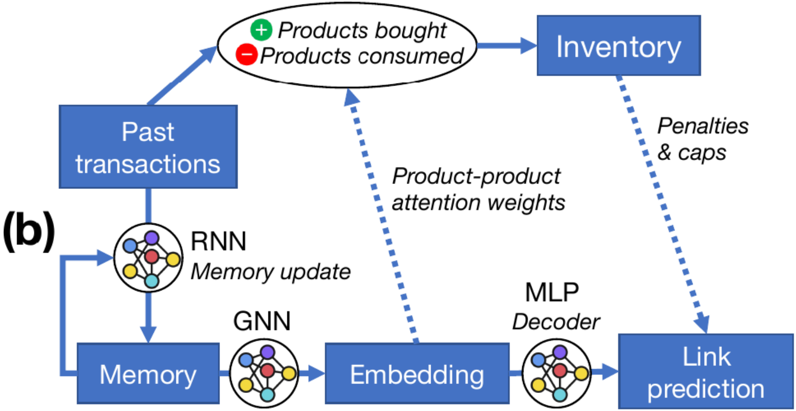
Módulo inventario

$$\alpha_{p_1 p_2} = \text{ReLU}\left(\mathbf{z}_{p_1} \mathbf{W}_{\mathrm{att}} \mathbf{z}_{p_2} + \nu_{p_1 p_2}\right) \tag{4}$$

$$\begin{aligned} \ell_{\text{inv}}(i,t) = & \lambda_{\text{debt}} \sum_{p \in [m]} \max\left(0,\ \text{consume}(i,p,t) - \mathbf{x}_i^{(t)}[p]\right) \\ & - \lambda_{\text{cons}} \sum_{p \in [m]} \text{consume}(i,p,t). \end{aligned} \tag{5}$$

$$\ell_{\text{inv}}(t) = \frac{1}{n} \sum_{i \in [n]} \ell_{\text{inv}}(i,t) + \lambda_{L_2} \sqrt{\sum_{p_1, p_2 \in [m]} \nu_{p_1 p_2}^2} \tag{6}$$

SC-TGN



Inventory + SC-TGN

SC-GraphMixer

- Simplicidad
- *Multi-layer perceptrons* - MLPs

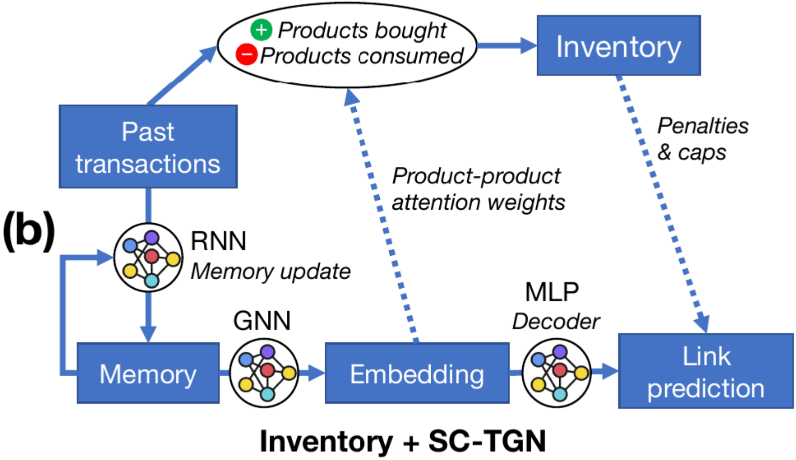
Decoder

$$\hat{y}(s,b,p,t) = \mathrm{DEC}\left(\mathbf{z}_s^{(t)}, \mathbf{z}_b^{(t)}, \mathbf{z}_p^{(t)}\right) = \mathrm{MLP}\left(\left[\mathbf{z}_s^{(t)} \middle| \mathbf{z}_b^{(t)} \middle| \mathbf{z}_p^{(t)}\right]\right) \tag{7}$$

$$\mathrm{pen}(s,b,p,t) = -\sum_{p' \in [m]} \max\left(0, \alpha_{pp'} - \mathbf{x}_s^{(t)}[p']\right) \tag{8}$$

$$\mathrm{cap}(s,b,p,t) = \min_{p' \in [m]; \alpha_{pp'} > 0} \left\{ \frac{\mathbf{x}_s^{(t)}[p']}{\alpha_{pp'}} \right\} \tag{9}$$

SC-TGN


Products bought
Products consumed
Inventory
Past transactions
Penalties & caps
Product-product attention weights
(b)
RNN
Memory update
GNN
MLP
Decoder
Memory
Embedding
Link prediction
Inventory + SC-TGN

Entrenamiento y evaluación

- **Aprendiendo funciones de producción**
- **Existencia de aristas y muestreo negativo**
- **Peso de la arista**

Datos de las cadenas de suministro

1. **Mundo real**
2. **Simulador** ***SupplySim***

Datos mundo real

- TradeSparq
- **Dataset automotriz** - Tesla
- **Dataset equipos industriales** - IED

Datos simulador *SupplySim*

- **Construyendo G_{prod}**
- **Construyendo relaciones proveedor-comprador**
- **Generando transacciones**

Experimentos

1. **Aprendiendo funciones de producción**
2. **Prediciendo futuras aristas**

Aprendiendo funciones de producción

1. **Correlaciones temporales**
2. **Punto de información mutua (PMI)**
3. **node2vec**

Aprendiendo funciones de producción

	SS-std	SS-shocks	SS-missing	IED
Random baseline	0.124 (0.009)	0.124 (0.009)	0.124 (0.009)	0.060 (0.002)
Temporal correlations	0.745	0.653	0.706	0.128
PMI	0.602	0.602	0.606	0.175
node2vec	0.280	0.280	0.287	0.127
Inventory module (direct)	0.771 (0.005)	0.770 (0.006)	0.744 (0.006)	0.143 (0.004)
Inventory module (emb)	**0.790** (0.005)	**0.778** (0.011)	**0.755** (0.007)	**0.262** (0.005)

Table 1: Results for production learning, evaluated with mean average precision (MAP $\uparrow$). For the models with randomness, we report mean and standard deviation (in parentheses) over 10 seeds.

Aprendiendo funciones de producción

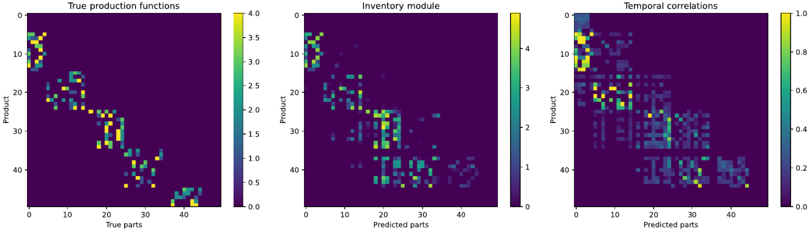


Figure 2: True production functions (left), predictions from inventory module (middle), predictions from temporal correlations (right), trained on SS-std.

Prediciendo futuras aristas

1. **Edgebank**
2. **Static**
3. **Graph transformer**

Prediciendo futuras aristas

	SS-std	SS-shocks	SS-missing	Tesla	IED
Edgebank (binary)	0.174	0.173	0.175	0.131	0.164
Edgebank (count)	0.441	0.415	0.445	0.189	0.335
Static	0.439 (0.001)	0.392 (0.002)	0.442 (0.001)	0.321 (0.001)	0.358 (0.001)
Graph transformer	0.431 (0.003)	0.396 (0.024)	0.428 (0.003)	0.507 (0.020)	0.613 (0.045)
SC-TGN	0.522 (0.003)	0.449 (0.004)	**0.494** (0.004)	**0.820 (0.007)**	**0.842 (0.004)**
SC-TGN+inv	**0.540** (0.003)	**0.461** (0.009)	0.494 (0.004)	0.818 (0.004)	0.841 (0.008)
SC-GraphMixer	0.453 (0.005)	0.426 (0.004)	0.446 (0.003)	0.690 (0.027)	0.791 (0.009)
SC-GraphMixer+inv	0.497 (0.004)	0.448 (0.004)	0.446 (0.002)	0.681 (0.014)	0.791 (0.008)
Edgebank (avg)	0.341	0.387	0.349	1.148	0.489
Static	0.343 (0.008)	0.425 (0.019)	0.374 (0.027)	1.011 (0.007)	0.504 (0.018)
Graph transformer	0.340 (0.005)	0.398 (0.025)	0.361 (0.016)	0.885 (0.024)	0.425 (0.008)
SC-TGN	**0.303** (0.003)	**0.359** (0.007)	0.313 (0.002)	0.796 (0.012)	0.428 (0.011)
SC-TGN+inv	0.312 (0.003)	0.370 (0.009)	**0.312** (0.002)	0.801 (0.015)	**0.422 (0.011)**
SC-GraphMixer	0.318 (0.003)	0.384 (0.005)	0.330 (0.005)	0.774 (0.077)	0.457 (0.008)
SC-GraphMixer+inv	0.320 (0.004)	0.378 (0.005)	0.328 (0.003)	**0.767 (0.054)**	0.454 (0.012)

Table 2: Results for edge prediction. Top 8 rows are edge existence, evaluated with mean reciprocal rank (MRR $\uparrow$). Bottom 7 rows are edge weight (i.e., transaction amount), evaluated with root mean squared error (RMSE $\downarrow$). We report mean and standard deviation (in parentheses) over 10 seeds.

Trabajo relacionado

- Otros sistemas
- Descubrimiento causal temporal
- Dominio: cadenas de suministro

Conclusión

- TPGs
- Nueva clase de GNNs
- Objetivos
- *SupplySim*
- Trabajo a futuro

Referencias

[1] [2]

Zhang S. Kang J. Yuan B. Wu H. Zhou X. Tong H. Cong, W. and M. Mahdavi.
Do we really need complicated model architectures for temporal networks?
In *Proceedings of the 11th International Conference on Learning Representations*, 2023.

Chamberlain B. Frasca F. Eynard D. Monti F. Rossi, E. and M. Bronstein.
Temporal graph networks for deep learning on dynamic graphs.
In *ICML 2020 Workshop on Graph Representation Learning*, 2020.

Apéndice

Detalles del modelo

1. **SC-TGN**
2. **SC-GraphMixer**
3. **Penalizaciones y límites del módulo de inventario**
4. **Entrenamiento y evaluación del modelo**

SC-TGN

- Memoria
- Función de mensaje
- Actualizador de memoria
- Incrustación
- Nuevos elementos en SC-TGN

Memoria

- $\mathbf{m}_i^{(t)}$
- $\mathbf{v}_i^{(0)}$

Función de mensaje

$$\mathrm{msg}_e = \left[\mathbf{m}_s^{(t)} \left| \mathbf{m}_b^{(t)} \right| \mathbf{m}_p^{(t)} \mid \mathrm{cnc}(t) \right] \tag{10}$$

Actualizador de memoria

$$\mathbf{m}_i^{(t+1)} = RNN\left(\mathbf{m}_i^{(t)}, \overline{msg}_i^{(t)}\right) \tag{11}$$

Incrustación

- $\mathbf{z}_i^{(t)}$
- Incrustación ID
- Modelo UniMP

Nuevos elementos en SC-TGN

- Hiperaristas
- Prediciendo pesos de aristas
- Actualización de penalización
- Memoria inicial aprendible $\mathbf{v}_i^{(0)}$
- Entrenando siguiendo el esquema de muetreo negativo

Nuevos elementos en SC-TGN

$$\ell_{\text{update}} = \frac{\lambda_{\text{update}}}{m+n} \sum_{i \in [m+n]} \left\| \mathbf{m}_i^{(t+1)} - \mathbf{m}_i^{(t)} \right\|_2 \tag{12}$$

SC-Graphmixer

- Codificador de enlaces
- Codificador de nodos
- Nuevos elementos SC-Graphmixer

Codificador de enlaces

- Codificación de tiempo
- Construcción matriz $\mathbf{T}_i(t)$
- MLP-mixer de una capa

Codificación de tiempo

Dado un paso de tiempo t, GraphMixer lo codifica a un vector de dimensión d. Utiliza $\omega = \left\{\alpha^{-(d'-1)/\beta}\right\}_{d'=1}^{d}$, donde α y β son hiperparámetros predefinidos, y proyecta t a $\cos(t \times \omega) \in [-1, +1]$.

Construcción matriz $\mathbf{T}_i(t)$

Cada fila corresponde una arista reciente de i, donde el enlace es representado por la concatenación de $\cos((t - t_e) \times \omega)$, donde t_e es el paso de tiempo de la arista, y las características del enlace. Codificar $t - t_e$ en vez de t_e captura con reciente fue la arista, y de esa forma cuanta influencia tiene sobre el paso de tiempo actual.

MLP-mixer de una capa

$$\mathbf{H}_{\text{token}} = \mathbf{T}_i(t) + \mathbf{W}^{(2)}_{\text{token}} \ \text{GeLU} \left(\mathbf{W}^{(2)}_{\text{token}} \ \text{Layer Norm} \left(\mathbf{T}_i(t) \right) \right) \tag{13}$$

$$\mathbf{H}_{\text{channel}} = \mathbf{H}_{\text{token}} + \text{GeLU} \left(\text{Layer Norm} \left(\mathbf{H}_{\text{token}} \right) W^{(1)}_{\text{channel}} \right) W^{(2)}_{\text{channel}} . \tag{14}$$

Codificador de nodos

$$\mathbf{s}_i(t) = \mathbf{x}_i^{\mathrm{node}} + \frac{1}{|\mathcal{N}(i; t - T, t)|} \sum_{j \in \mathcal{N}(i; t-T, t)} \mathbf{x}_j^{\mathrm{node}} \tag{15}$$

Nuevos elementos SC-Graphmixer

- Hiperaristas
- Características aprendibles del nodo
- Prediciendo pesos de aristas
- Entrenando siguiendo el esquema de muetreo negativo

Penalizaciones y límites del módulo de inventario

- Opcionalidad
- Solo puede ayudar
- Limitaciones

Entrenamiento y evaluación del modelo

- Pérdida del modelo
- Aprendiendo funciones de producción
- Prediciendo existencia de aristas
- Prediciendo peso de las aristas

Pérdida del modelo

$$\mathcal{L} = \ell_{\text{exist}} + \ell_{\text{weight}} + \ell_{\text{inv}} + \ell_{\text{update}} \tag{16}$$

Aprendiendo funciones de producción

$$\mathrm{AvePrec}(p_o) = \frac{1}{|\mathcal{P}_{p_o}|} \sum_{k=1}^{m} \mathrm{Prec} \mathbin{@} \mathrm{K}(\alpha_{p_o}, \mathcal{P}_{p_o}, k) \cdot \mathbb{1}[\mathrm{pos}(\alpha_{p_o}, k) \in \mathcal{P}_{p_o}] \tag{17}$$

$$\mathrm{Prec} \mathbin{@} \mathrm{K}(\alpha, \mathcal{P}, k) = \frac{1}{k} \sum_{k'=1}^{k} \mathbb{1}\left[\mathrm{pos}\left(\alpha, k'\right) \in \mathcal{P}\right] \tag{18}$$

Prediciendo existencia de aristas

$$MRR = \frac{1}{|B|} \sum_{e \in B} \left(\frac{\sum_{n \in \mathcal{N}_e} \mathbb{1}\left[\hat{y}_n < \hat{y}_e\right] + \sum_{n \in \mathcal{N}_e} \mathbb{1}\left[\hat{y}_n \leq \hat{y}_e\right]}{2} + 1 \right)^{-1} \tag{19}$$

Prediciendo peso de las aristas

$$RMSE = \sqrt{\frac{1}{|B|}\sum_{e \in B}(amt(e) - \hat{y}_e)^2} \tag{20}$$

Datos

1. **Datos mundo real**
2. **Detalles acerca de *SupplySim***

Datos mundo real

- **Dataset automotriz**
- **Dataset equipos industriales**
- **Limitaciones**

Datos mundo real

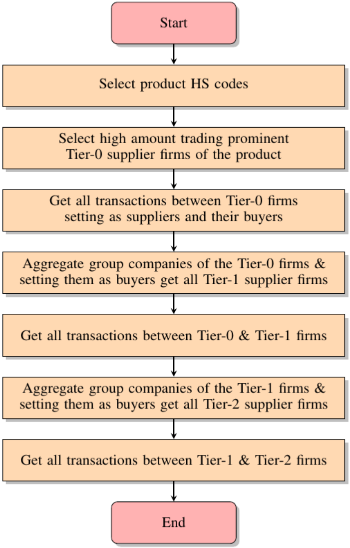


Figure 3: Overview of supply chain network data construction process.

Datos mundo real

Attribute Counts	SS-std	Tesla	IED
# Product Nodes	50	2,690	3,029
# Firms Nodes	119	11,628	2,583
# Transactions	71646	581,002	279,712
Timespan (Days)	198	1683	359

Table 3: Dataset statistics.

Detalles acerca de *SupplySim*

1. **Grafos estáticos**
2. **Generando transacciones variables en el tiempo**

Grafos estáticos

- **Construyendo G_{prod}**
- **Construyendo relaciones proveedor-comprador**

Construyendo G_{prod}

- Nivel 0: producto 0 al producto $n_{\text{exog}} - 1$,
- Nivel 1: producto n_{exog} al producto $n_{\text{exog}} + n_{\text{tier}} - 1$,
- Nivel 2: producto $n_{\text{exog}} + n_{\text{tier}}$ al producto $n_{\text{exog}} + 2 \cdot n_{\text{tier}} - 1$,
- . . .
- Nivel T : producto $n_{\text{exog}} + (T - 1) \cdot n_{\text{tier}}$ al producto $n_{\text{exog}} + T \cdot n_{\text{tier}} - 1$,
- Nivel $T + 1$: producto $n_{\text{exag}} + T \cdot n_{\text{tier}}$ al producto $n_{\text{exog}} + T \cdot n_{\text{tier}} + n_{\text{consumer}} - 1$.

Construyendo G_{prod}

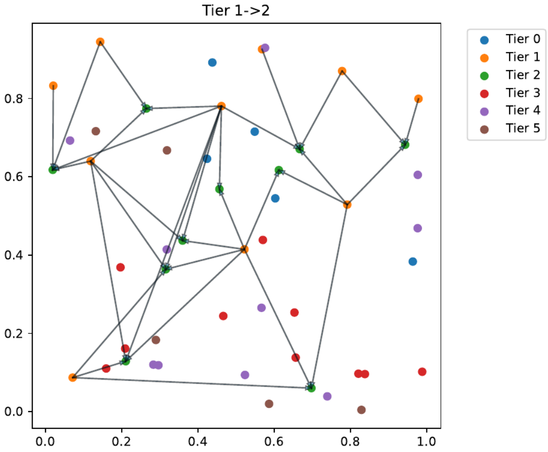


Figure 4: Visualizing products in our synthetic datasets. Each point represents the position of one of the 50 products, and points are color-coded by the product's tier. We also denote part-product relations between Tier 1 and Tier 2 products, where an arrow from product p_1 to product p_2 means that p_1 is required to make p_2. For each product, we sample its number of parts from $\{1, 2, 3, 4\}$ uniformly, then assign its parts to the closest products in the previous tier, resulting in commonly co-occurring parts.

Construyendo relaciones proveedor-comprador

- Grupo 0: Nivel 0 al Nivel $n_{\text{consec}} - 1$,
- Grupo 1: Nivel 1 al Nivel n_{consec},
- Grupo 2: Nivel 2 al Nivel $n_{\text{consec}} + 1$,
- $\ldots$
- Grupo $T - n_{\text{consec}} + 2$: Nivel $T - n_{\text{consec}} + 2$ al Nivel $T + 1$.

Generando transacciones variables en el tiempo

- **Modelo ARIO**
- **Nueva oferta**
- **Producción**
- **Nueva demanda**
- **Productos nivel 0 y oferta exógena**
- **Productos nivel final y demanda exógena**

Nueva demanda

$$k_{f,p_i}^{(t)} = \sum_{(b,f,p,k)\in\mathcal{I}^{(t)}} u_{p_i p} \cdot k - \sum_{(f,s,p_i,k)\in\mathcal{I}^{(t)}} k \qquad (21)$$

Productos nivel 0 y oferta exógena

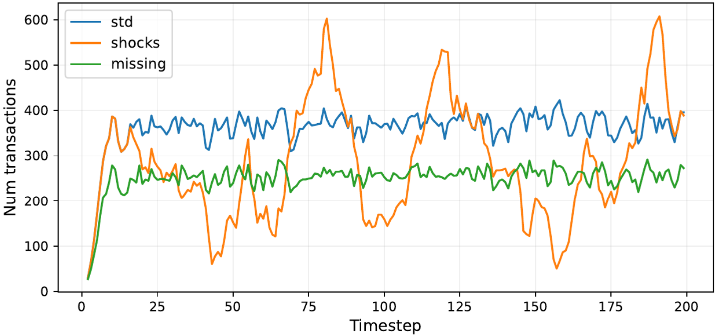


Figure 5: Visualizing the number of transactions per timestep, over the three synthetic datasets: SS-std, SS-shocks, and SS-missing.

Productos nivel final y demanda exógena

$$\nu \sim \mathcal{N}(0, 0.1) \tag{22}$$

$$\lambda = \begin{cases} 1 \text{ if type} & = \text{ uniform}, \\ 2 \text{ if type} & = \text{ weekday and } t \bmod 7 < 5, \\ 0.5 \text{ if type} & = \text{ weekday and } t \bmod 7 \geq 5, \\ 0.5 \text{ if type} & = \text{ weekend and } t \bmod 7 < 5, \\ 2 \text{ if type} & = \text{ weekend and } t \bmod 7 \geq 5. \end{cases} \tag{23}$$

$$d(p, t) = \lambda \cdot (d(p, t - 1) + \nu) \tag{24}$$

$$d(f, p, t) = \text{Poisson}(d(p, t)) \tag{25}$$

Detalles acerca de los experimentos

1. **Aprendizaje de producción**
2. **Predicción de aristas**

Aprendizaje de producción

- **Correlaciones temporales**
- **Punto de información mutua (PMI)**
- **node2vec**
- **Módulo de inventario**

Punto de información mutua (PMI)

$$\mathrm{PMI}\left(p_1, p_2\right) = \log\left(\frac{\Pr\left(\mathrm{buy}\left(p_1\right) \wedge \mathrm{supply}\left(p_2\right)\right)}{\Pr\left(\mathrm{buy}\left(p_1\right)\right) \cdot \Pr\left(\mathrm{supply}\left(p_2\right)\right)}\right) \quad (26)$$

Módulo de inventario

	Synthetic data	Tesla	IED
SC-TGN			
Memory dimension	500	1000	1000
Embedding dimension	500	1000	1000
Time dimension	100	100	100
# neighbors for node embedding	20	20	100
Update penalty λ_{update}	1	1	1
SC-GraphMixer			
# MLPMixer layers	2	2	2
Node encoding dimension	500	50	300
Link encoding dimension	100	10	10
# neighbors for node encoding	20	100	10
# neighbors for link encoding	20	10	2
Inventory module			
Debt penalty λ_{debt}	5	5	5
Consumption reward λ_{cons}	4	4	4
Adjustment penalty λ_{adjust}	4	4	4
Training parameters			
Batch size	30	30	100 (SC-TGN), 30 (SC-GraphMixer)
Learning rate	0.001	0.001	0.001
Max # epochs	100	100	100
Patience	10	10	10

Table 4: Hyperparameters that we used in our experiments.

Módulo de inventario

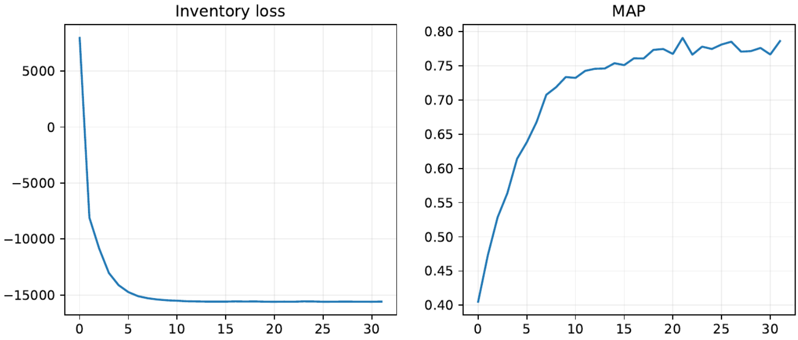


Figure 6: Comparing inventory module's loss (6) vs. MAP on ground-truth production functions, trained on SS-std.

Predicción de aristas

- **Entrenamiento**
- **Ablaciones**
- **Análisis de experimentos +inv**

Entrenamiento

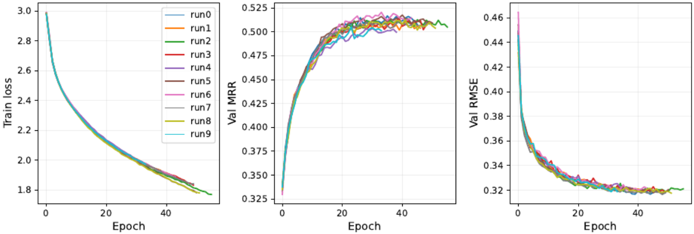


Figure 7: Performance over 10 random seeds of SC-TGN on SS-std.

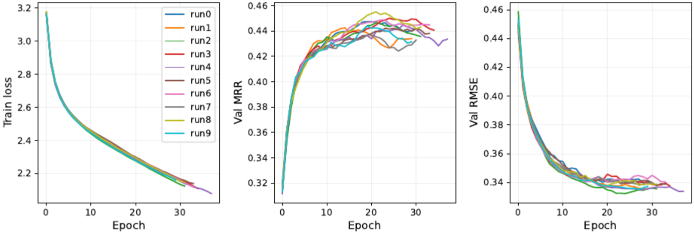


Figure 8: Performance over 10 random seeds of SC-GraphMixer on SS-std.

Ablaciones

	Tesla	IED
TGN	0.612 (0.009)	0.582 (0.016)
SC-TGN (id)	0.537 (0.021)	0.422 (0.015)
SC-TGN	**0.820** (0.007)	**0.842** (0.004)

Table 5: Ablations of SC-TGN: comparing to original TGN (Rossi et al., 2020) and SC-TGN with ID embeddings, i.e., use memory directly as embedding, instead of applying GNN to memories. We only evaluate edge existence here, with mean reciprocal rank (MRR $\uparrow$), and leave out edge weight, since the original TGN did not predict edge weight. We report mean and standard deviation (in parentheses) over 10 seeds.

Análisis de experimentos +inv

	SS-std	SS-shocks	SS-missing
SC-TGN	0.522 (0.003)	0.449 (0.004)	**0.494** (0.004)
SC-TGN+inv*	**0.548** (0.003)	**0.474** (0.003)	0.476 (0.003)
SC-GraphMixer	0.453 (0.005)	0.426 (0.004)	0.446 (0.003)
SC-GraphMixer+inv*	0.477 (0.005)	0.450 (0.005)	0.430 (0.003)

Table 6: Testing the impact of inventory module on edge existence prediction, when the inventory module is provided the ground-truth production functions. We report mean and standard deviation (in parentheses) over 10 seeds.